



Vinícius Costa Villas Bôas Segura

**BONNIE: Building Online Narratives from
Noteworthy Interaction Events**

Tese de Doutorado

Thesis presented to the Programa de Pós-Graduação em
Informática of the Departamento de Informática, PUC-Rio as
partial fulfillment of the requirements for the degree of Doutor
em Ciências – Informática.

Advisor: Prof^ª. Simone Diniz Junqueira Barbosa

Rio de Janeiro
September 2016



Vinícius Costa Villas Bôas Segura

**BONNIE: Building Online Narratives from
Noteworthy Interaction Events**

Thesis presented to the Programa de Pós-Graduação em
Informática of the Departamento de Informática do Centro
Técnico Científico da PUC-Rio as partial fulfillment of the
requirements for the degree of Doutor.

Prof^a. Simone Diniz Junqueira Barbosa

Advisor

Departamento de Informática — PUC-Rio

Prof. Hélio Côrtes Vieira Lopes

Departamento de Informática — PUC-Rio

Prof. Alberto Barbosa Raposo

Departamento de Informática — PUC-Rio

Prof. Renato Fontoura de Gusmão Cerqueira

IBM Research — Brazil

Prof^a. Milene Selbach Silveira

PUCRS

Prof. Márcio da Silveira Carvalho

Coordinator of the Centro Técnico Científico da PUC-Rio

Rio de Janeiro, September 26th, 2016

All rights reserved.

Vinícius Costa Villas Bôas Segura

Vinícius Segura holds a M.Sc. degree in HCI and a B.Sc. in Computer Engineering, both from PUC-Rio. During his masters, he developed pen-based early prototyping software that allows users to sketch user interfaces and program behaviors. He is a Research Staff Member at IBM Research in Rio de Janeiro, Brazil, working with the Visual Analytics and Comprehension Research Group. Prior to joining IBM Research, Vinícius worked at the Computer Graphics Technology Group at PUC-Rio (Tecgraf). His main research interests are human-computer interaction, computer graphics, and information visualization.

Bibliographic data

Segura, Vinícius Costa Villas Bôas

BONNIE: Building Online Narratives from Noteworthy Interaction Events / Vinícius Costa Villas Bôas Segura ; advisor: Simone Diniz Junqueira Barbosa. – 2016.

154 f. : il. color ; 30 cm

Tese (Doutorado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2016.

Inclui bibliografia

1. Informática – Teses. 2. Histórico de interação. 3. Log de software. 4. Visualização de histórico. 5. Narrativa. 6. Análise visual. I. Barbosa, Simone Diniz Junqueira. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

Acknowledgments

To CNPq, for funding my research.

To Simone Barbosa, for guiding me through the academic “trilogy.”

To my family, for the support and positive thinking even from afar.

To Kita, for being by my side through all these thriller times.

To Bonnie Tyler, for all the corginess and naming inspiration.

Abstract

Segura, Vinícius Costa Villas Bôas; Barbosa, Simone Diniz Junqueira (advisor). **BONNIE: Building Online Narratives from Noteworthy Interaction Events**. Rio de Janeiro, 2016. 154p. D.Sc. Thesis — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Nowadays, we have access to data of unprecedentedly large size, high dimensionality, and complexity. To extract unknown and unexpected information from such complex and dynamic data, we need effective and efficient strategies. One such strategy is to combine data analysis and visualization techniques, which is the essence of visual analytics applications. After the knowledge discovery process, a major challenge is to filter the essential information that led to a discovery and to communicate the findings to other people. We propose to take advantage of the trace left by the exploratory data analysis, in the form of user interaction history, to aid in this process. With the trace, the user can choose the desired interaction steps and create a narrative, sharing the acquired knowledge with readers. To achieve our goal, we have developed the BONNIE (Building Online Narratives from Noteworthy Interaction Events) framework. The framework comprises a log model to register the interaction events, auxiliary code to help the developer instrument his or her own code, and an environment to view the user's own interaction history and build narratives. This thesis presents our proposal for communicating discoveries in visual analytics applications, the BONNIE framework, and a few empirical studies we conducted to evaluate our solution.

Keywords

Interaction history; Software log; History visualization; Narrative; Visual analytics;

Resumo

Segura, Vinícius Costa Villas Bôas; Barbosa, Simone Diniz Junqueira. **BONNIE: Construindo narrativas online a partir de eventos de interação relevantes**. Rio de Janeiro, 2016. 154p. Tese de Doutorado — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Nos dias de hoje, temos acesso a dados de tamanho, dimensionalidade e complexidade sem precedentes. Para extrair informações desconhecidas e inesperadas desses dados complexos e dinâmicos, necessitamos de estratégias efetivas e eficientes. Uma dessas estratégias é usar aplicações de análise visual (*visual analytics*), que combinam técnicas de análise de dados e de visualização. Depois do processo de descoberta de conhecimento, um grande desafio é filtrar a informação essencial que levou à descoberta e comunicar os achados a outras pessoas. Nós propomos tirar proveito do traço deixado pela análise exploratória de dados, sob a forma do histórico da interação do usuário, para ajudar nesse processo. Com o traço, o usuário pode escolher os passos de interação desejados e criar uma narrativa, compartilhando o conhecimento adquirido com os leitores. Para atingir nosso objetivo, desenvolvemos o arcabouço BONNIE (Building Online Narratives from Noteworthy Interaction Events – Construindo Narrativas Online a partir de Eventos de Interação Relevantes). O arcabouço compreende um modelo de log para registrar os eventos de interação, código auxiliar para ajudar o(a) desenvolvedor(a) a instrumentar o seu próprio código, e um ambiente para visualizar o histórico de interação e construir narrativas. Esta tese apresenta nossa proposta para comunicar descobertas em aplicações de análise visual, o arcabouço BONNIE, e alguns estudos empíricos que realizamos para avaliar nossa solução.

Palavras-chave

Histórico de interação; Log de software; Visualização de histórico; Narrativa; Análise visual;

Contents

1	Introduction	13
1.1	Research Goals and Methodology	16
1.2	Thesis Structure	17
2	Foundations	18
2.1	Why Visual Analytics Applications?	18
2.2	Why Narratives?	23
3	BONNIE	26
4	Log Model	32
5	SrcSys Instrumentation	39
5.1	Keep Definitions Updated	39
5.2	Log Interaction Events	41
5.3	Use Compatible Data Services	43
5.4	Use Compatible Visualization Components	43
6	User Interaction History Visualization	45
6.1	History Visualization Analytical Study with PoN and CDN	49
6.2	History Visualization User Study	52
7	Narrative Builder	55
7.1	Narrative Builder User Study	58
8	Related Work	65
8.1	Visual Analytics	65
8.2	Annotating Visualizations	67
8.3	Data Narratives	69
8.4	Concluding Remarks	74
9	Conclusion	75
9.1	Contributions	75
9.2	Future Work	76
10	References	80
A	Visual Analytics Definitions	89
B	PROV-XML Example	91
B.1	Definition Hierarchy	91
B.2	Interaction Definition	92
C	Definitions Registration Example	99
D	History Visualization Analytical Study in Detail	102

D.1	Evaluation Using Physics of Notation	103
D.2	Evaluation Using Cognitive Dimensions of Notation	109
D.3	Main Impacts	114
E	History Visualization User Study Details	116
E.1	Study Material	117
E.2	Methodology	124
E.3	Study Results	126
F	Narrative Builder User Study Details	138
F.1	Study Material	139
F.2	Methodology	139
F.3	Study Results	147
G	Backlog	153

List of Figures

1.1	Data growth estimation in 2012.	14
1.2	Data generated in one minute	15
1.3	Methodology summary diagram.	17
2.1	Word cloud from <i>visual analytics</i> definitions found in appendix A.	19
2.2	A model of the interactive visualization process.	20
2.3	Visual variables proposed by Bertin, Morrison, and MacEachren.	20
2.4	Example of common interaction techniques.	21
2.5	Examples of Gestalt principles of perception.	22
3.1	BONNIE interaction sequence.	27
3.2	BONNIE's framework considered architecture.	28
3.3	BONNIE's main UI.	29
3.4	WISE's main UI, highlighting its components.	31
4.1	BONNIE's log model relations.	33
4.2	BONNIE's log model showing the attributes of each element.	34
4.3	Multi-level typology of abstract visualization tasks.	36
4.4	BONNIE's log model as expressed using PROV-DM.	38
6.1	User interaction history visualization.	46
6.2	Example of GIT commit graph.	47
6.3	Example of a navigation row.	47
6.4	Examples of action rows.	48
6.5	Nodes representing visualization effects.	48
6.6	Example of effect row.	49
6.7	Line colors detail.	49
6.8	Action row comment feature.	50
6.9	Effect row metadata information feature.	50
6.10	BONNIE interaction sequence considering the different players involved and their communication.	53
7.1	Adding a panel to the narrative.	56
7.2	Adding a textual element to a panel.	57
7.3	Adding a visualization element to a panel.	57
7.4	Layout of textual and visualization elements.	58
7.5	A single panel narrative.	58
7.6	Distribution of answers to history visualization questionnaire, considering both tasks and previous study.	61
7.7	Distribution of answers to TAM questionnaire grouped by constructs and considering tasks individually.	62
7.8	Distribution of answers to TAM questionnaire grouped by constructs, aggregating answers for both tasks.	63
8.1	VizDeck dashboard interface.	66
8.2	The sense.us collaborative visualization system.	67

8.3	Many Eyes interface.	68
8.4	A visualization produced by Contextifier with some callouts to interface features.	69
8.5	Telling a story using SketchStory.	70
8.6	The Ellipsis interface.	71
8.7	Example of Tableau interface.	72
8.8	VisTrails interface with highlights.	73
9.1	CATS architecture.	79
D.1	History visualization iteration for the analytical study.	102
D.2	Alternate representation for session.	104
D.3	Session and page context as user scrolls through the visualization.	107
D.4	Example of using different symbols for the different visualization tasks.	108
D.5	History visualization concepts not explained to users.	111
D.6	Multiple nodes collapsed into a single action row.	112
D.7	Expanded action row shows different tasks.	112
D.8	Each line represents a visualization component.	113
D.9	User actions versus application actions.	113
D.10	History visualization analytical study main impacts.	114
E.1	History visualization iteration for the user study.	116
E.2	Profile questionnaire, page 1.	118
E.3	Profile questionnaire, page 2.	119
E.4	Study script, page 1.	120
E.5	Study script, page 2.	121
E.6	Study script, page 3.	122
E.7	Final questionnaire.	123
E.8	Norman's seven stages of action.	133
E.9	Distribution of answers to the final questionnaire.	136
F.1	BONNIE help, page 1.	140
F.2	BONNIE help, page 2.	141
F.3	BONNIE help, page 3.	142
F.4	Study script, page 1.	143
F.5	Study script, page 2.	144
F.6	History visualization questionnaire.	145
F.7	TAM questionnaire, combining statements from TAM and TAM2.	146

List of Tables

2.1	Narrative visualization perspectives spectrum.	24
2.2	Narrative visualization genres comparison.	25
4.1	Descriptions of visualization tasks.	37
D.1	Mapping from log model to our visual notation.	104
D.2	Visual variables, their power, and capacity.	108
D.3	Cognitive dimensions for the CDN.	110
E.1	Profile questionnaire results.	127
E.2	Task 1 results.	128
E.3	Summary of task 1 results.	128
E.4	Task 2 results.	129
E.5	Summary of task 2 results.	130
E.6	Task 3 results.	130
E.7	Task 4 results.	131
E.8	Task 5 results.	132
E.9	Final questionnaire answers.	135
F.1	Association between TAM/TAM2 constructs and the questions.	147
F.2	Task 1 results.	148
F.3	Task 1 history visualization questionnaire answers by participant.	149
F.4	Task 1 TAM questionnaire answers by participant.	150
F.5	Task 2 history visualization questionnaire answers by participant.	151
F.6	Task 2 TAM questionnaire answers by participant.	152

*To invent your own life's meaning is not easy,
but it's still allowed, and I think you'll be
happier for the trouble.*

Bill Watterson, commencement speech at Kenyon College (May 20, 1990).

1 Introduction

Nothing – not the careful logic of mathematics, not statistical models and theories, not the awesome arithmetic power of modern computers – nothing can substitute here for the flexibility of the informed human mind.

John W. Tukey & Martin B. Wilk, *Data Analysis & Statistics (1966)*

Internet. Mobile phones. Wearable devices. Internet of things. We are living in a world where data are constantly being produced and consumed, growing considerably each year (as shown in figure 1.1), at around 40% compound annual rate, expecting to reach nearly 45 zettabytes¹ in 2020.²

In 2010 alone, 1200 exabytes³ of data were generated, the equivalent of the Library of Congress's content – times 60 million (Heer et al., 2010). Two years later, in 2012, in every two days we generated as much data as in all of human history up to 2003 (IBM, 2012, p. 5). In 2013, we generated about 2.5 exabytes of data every day (IBM, 2013, p. 13).

Figure 1.2 highlights the amount of data generated in a minute on popular websites in 2012, 2014, and 2015. If we take just Youtube as an example, in 2012 users uploaded about 48 hours of video in a single minute. Two years later, this number increased to 72 hours. One year later, there was a major bump, to 300 hours followed by another increase to 400 hours in the next year. This means that, in 2016, about 65.6 years of video were uploaded every single day.

Hence, it should come as no surprise that data-related topics have been trending in recent years. The term **big data** is commonly used to highlight the growth and availability of both structured and unstructured data. One of the most common definitions was articulated by Doug Laney back in 2001⁴ and is known as the “three V’s”: the increase in data volume, velocity, and variety.

¹1 zettabyte \approx 1000 exabytes \approx 1 billion terabytes

²https://www.atkearney.com/strategic-it/ideas-insights/article/-/asset_publisher/LCcgDeS4t85g/content/big-data-and-the-creative-destruction-of-today-s-business-models/10192

³1 exabyte \approx 1 million terabytes

⁴<http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf>

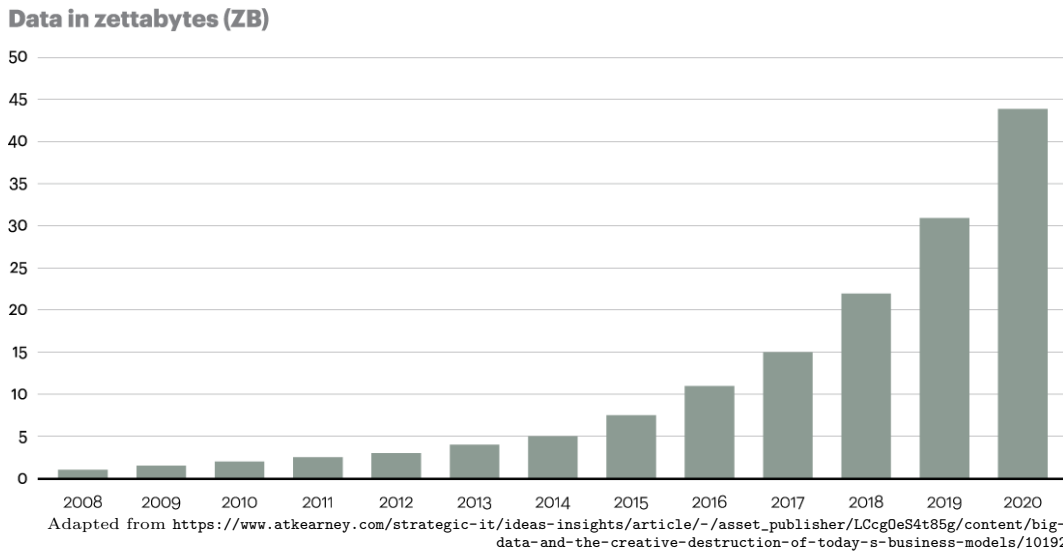


Figure 1.1: Data growth estimation in 2012.

Different authors added other dimensions – such as veracity, variability, and complexity⁵–, but the basic definition still stands.

It is such a broad term that after appearing for four years (since 2011) in Gartner’s hype cycle for emerging technologies, it was dropped in the 2015 edition.⁶ Betsy Burton, the report’s author, explained: “... big data has quickly moved over the Peak of Inflated Expectations and has become prevalent in our lives across many hype cycles. So big data has become a part of many hype cycles.”⁷

Research-wise, the bottleneck has shifted from data *acquisition* (when there are poor datasets) to data *analysis* (what to do with the recently available rich datasets) (Key et al., 2012). Human attention is now the limited resource. More recent definitions of *big data* already contemplate this change. For example, Gartner’s current *big data* definition is “high-volume, -velocity and -variety information assets that demand *cost-effective, innovative forms of information processing for enhanced insight and decision making*.”⁸

Effective and efficient strategies are needed to extract unknown and unexpected information from these data of unprecedentedly large size, high dimensionality, and complexity (Mennis & Guo, 2009). Only a combination of data analysis and visualization techniques can handle this complex and dynamic data (Keim et al., 2008). This combination is the basis of visual analytics, which aims to explore the best interplay of computers’ analytical

⁵http://www.sas.com/en_us/insights/big-data/what-is-big-data.html

⁶<http://www.gartner.com/newsroom/id/3114217>

⁷<http://www.datanami.com/2015/08/26/why-gartner-dropped-big-data-off-the-hype-curve/>

⁸<http://www.forbes.com/sites/gartnergroup/2013/03/27/gartners-big-data-definition-consists-of-three-parts-not-to-be-confused-with-three-vs/>

capabilities and users' cognitive ones.

After the knowledge discovery process, a major challenge is to filter the essential information that led to the discovery and to communicate the findings to others. To share the obtained knowledge, we can wield the power of a story. Besides transmitting information, stories are means to communicate contextual information and connect the author with the audience (Quesenbery & Brooks, 2010, page 19).

1.1 Research Goals and Methodology

The main goal of this work is to support users in creating data stories based on their interaction with a visual analytics application (VAApp). We hypothesize it would be useful to take advantage of the trace left by the exploratory data analysis, in the form of user interaction history. Our main research question is *“How can we provide support for users to communicate their findings based on their interaction history with a VAApp?”*

To tackle the problem at hand, we created more specific research “sub-questions”:

1. *What should we capture and how to save interaction events from a VAApp?*
2. *How should we change the VAApp to make it compatible with our solution?*
3. *How can we display the user's interaction history with another VAApp?*
4. *How to enable the user to create a narrative from his or her interaction history?*

Our methodology started with a proposal for a model to log interaction events considering a web VAApp. With this model, we instrumented a VAApp so we could validate our approach. Then we developed a visual representation for the log data. We established a visual notation and evaluated it with an analytical study followed by a user study. Finally we developed the narrative builder environment, evaluating it with a second user study. A summary of the methodology can be seen in figure 1.3. Despite the linearity of the diagram, the actual development had several iterations in each phase and some back-and-forth between phases (as represented by the dashed arrows). Additionally, the literature review is not included in the figure because it permeated every phase.

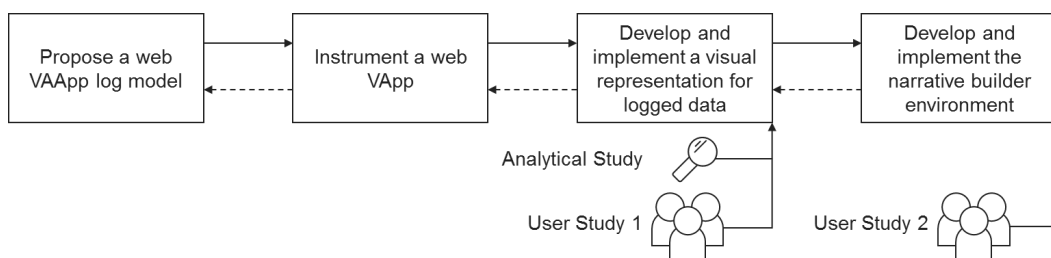


Figure 1.3: Methodology summary diagram.

1.2 Thesis Structure

We begin this thesis discussing the visualization and communication facets of this problem (chapter 2). Chapter 3 gives an overview of our framework, describing how we envision it integrating with and adding value to the regular VApp interaction workflow. The next four chapters relate to the aforementioned research “sub-questions”:

1. Chapter 4 introduces the developed log model to save the VApp’s interaction events.
2. Chapter 5 describes the changes that the VApp developer should implement in order to make the VApp compatible with our system.
3. Chapter 6 presents our interaction history visualization notation and two studies (an analytical one and another one with users) to evaluate the visualization component of our system.
4. Chapter 7 details how our system supports creating a narrative from the interaction events and a user study that evaluated this task.

The following chapter (chapter 8) compares our solution to some related works. Finally, chapter 9 concludes this thesis with some final remarks regarding the contributions of our work and future directions.

2 Foundations

The most efficient constructions are those in which any question, whatever its type and level, can be answered in a single instant of perception, that is, in a single image.

Bertin, J. & Berg, W, *Semiology of Graphics: Diagrams, Networks, Maps (2011)*

Our solution is focused on interacting with visual analytics applications (VAApps) and on generating a narrative afterwards from the interaction history. The next sections provides the motivation behind this scope, discussing why focus on VAApps (section 2.1) and narratives (section 2.2).

2.1 Why Visual Analytics Applications?

Visual analytics is an emerging and inherently multi-disciplinary research topic. It involves multiple processes, and can be applied in various different areas. The “common denominator” found in all definitions is that its main goal is to make use of a vast amount of data by combining the strengths of computers and humans, while complementing and enhancing the capabilities of each other (Kohlhammer et al., 2011; Andrienko et al., 2011). It is no surprise that, when analyzing the word frequency in those definitions¹ (represented as a word cloud in figure 2.1), “information”, “analysis”, “visualization”, “human”, and “capabilities” are the most frequent words.

On the one hand, computers can provide intelligent data analysis (Kohlhammer et al., 2011) without cognitive biases² (Green et al., 2008). Their enormous processing power (Aigner et al., 2007a) and superior working memory (Green et al., 2008) guarantee an incomparable mathematical, algebraic, and statistical prowess to handle massive volumes of data.

On the other hand, human users can contribute with their analytical capabilities and inherent visual perception (Kohlhammer et al., 2011). Together, these characteristics enable humans to efficiently perform visual information

¹Appendix A lists a series of definitions from the literature.

²The computers are unbiased by themselves. However, we can argue that developers’ biases pervade the computer behavior.

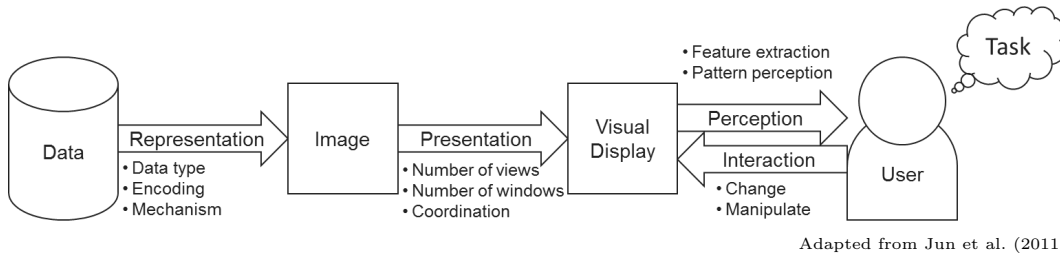


Figure 2.2: A model of the interactive visualization process.

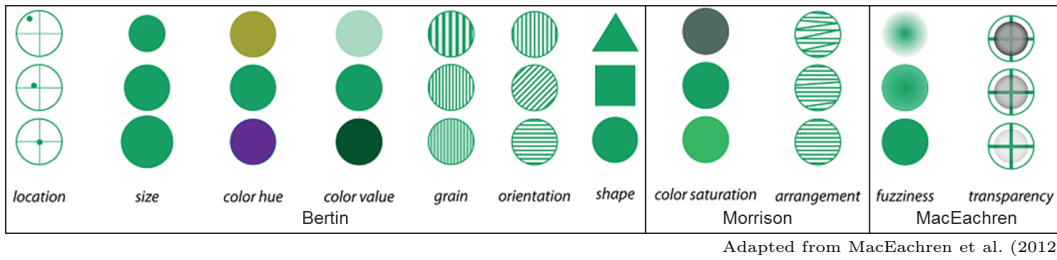


Figure 2.3: Visual variables proposed by Bertin, Morrison, and MacEachren.

try to determine the best visualization for a given dataset and a problem at hand (Heer et al., 2010; de Sousa & Barbosa, 2012, 2013; Mackinlay et al., 2007; Mackinlay, 1986). After choosing the visualization, an effective encoding should be used. Several guidelines can be found in literature, such as Bertin’s (1983) famous visual variables: location, size, color hue, color value, grain, orientation, and shape. Figure 2.3 illustrates his visual variables and some additional ones proposed by different authors.

The second step is **presentation**: how the chosen visualizations will appear on the display, the layout, and the available interactivity. Considering the available space, we should decide which representation should be used (*e.g.*, a single type or multiple representations) and how many views (*e.g.*, a single window or multiple windows).

Usually, to make the most of available screen space, some interaction techniques are used to coordinate visualizations and encode more data. Cockburn et al. (2008) review many interfaces according to these most common interaction techniques:

1. **Overview+detail** — Simultaneously displays an overview and a detailed view. The views are, therefore, spatially separated.
2. **Zooming** — The same view space is used to display both the overview

faceted scientific data), <http://idav.ucdavis.edu/~ki/STAR/> (performance), <http://dataphys.org/list/> (physical), <http://www.cvast.tuwien.ac.at/~alsallakh/SetViz/literature/www/index.html> (sets), <http://spacetimecubevis.com/> (temporal data), <http://textvis.lnu.se/> (text), <http://survey.timeviz.net/> (time), <http://vcg.informatik.uni-rostock.de/~hs162/treeposter/poster.html> (tree).

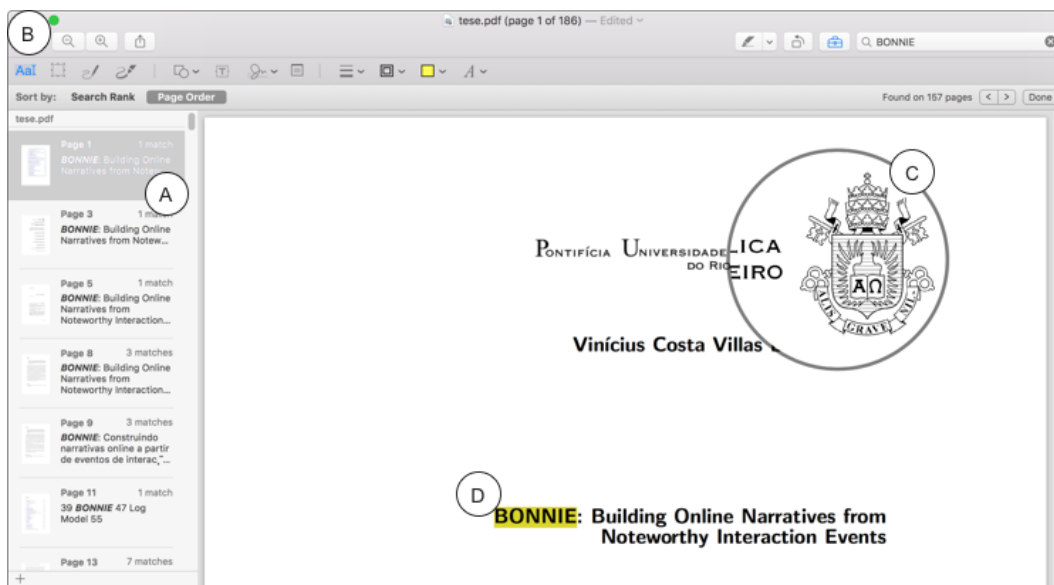


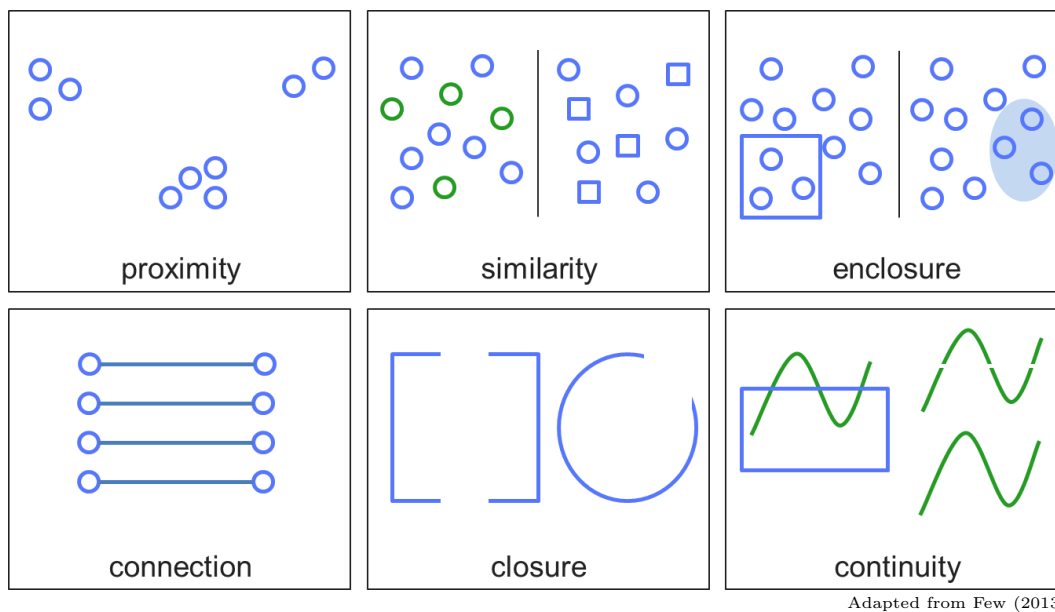
Figure 2.4: Example of common interaction techniques.

and detailed view. The views are separated in time, as the user zooms in/out.

3. ***Focus+context*** — The same view space is used to display both the overview and detailed view, at the same time. The detailed view (focus) is seamlessly integrated with the overview (context), usually applying some kind of distortion (*e.g.*, “fisheye”, scale modification).
4. ***Cue-based techniques*** — The objects can be rendered differently to attract focus and symbols can be introduced in the view to indicate off-view objects.

These techniques can be combined in the same application depending on the task being performed. We can find examples of these techniques in general-purpose applications, such as Apple’s “Preview”, the default PDF viewer in Mac computers. Figure 2.4 shows this application after the user searches for a term in the PDF. The left sidebar (A) shows the thumbnails of the pages which contain the search term (“BONNIE” in the example), giving an overview of results and the details (the page) on the main space (*overview+detail*). The user can zoom in or out (B) in the main view (*zooming*). Also, he or she can add a lens (C) which magnifies only a portion of the image (*focus+context*). Finally, all search results are rendered with a yellow highlight (D), attracting the user’s attention (*cue-based techniques*).

Once the data is displayed, the communication loop with the user starts. Pinker (1990) attributes to ***perception*** a chart’s effectiveness as a communication tool. *Perception* can be subdivided into two main components:



Adapted from Few (2013)

Figure 2.5: Examples of Gestalt principles of perception.

feature extraction and *pattern perception*. *Feature extraction* processes are driven by the sensory information and occur in parallel. For example, different colors can be detected almost instantaneously and in parallel. *Pattern perception* combines sensory information with our knowledge, expectations, and goals to determine which features can be grouped, how they can be associated with objects, and which objects are recognizable. One example of *pattern perception* can be seen in the *Gestalt principles of perception*: descriptions of visual behavior that explain how we perceive pattern, form, and organization in what we see (Few, 2013). These principles are illustrated in figure 2.5.

The other part of the communication loop is the user's *interaction* with the visualization. This allows the user to change and manipulate data presentation. Direct manipulation also enhances user control, leading to a more comprehensible, predictable, and controllable interface (Shneiderman, 1997). Interactive dynamics may include (Heer & Shneiderman, 2012): navigation, filtering, sorting, selecting, etc.

Finally, the *task* relates to the user activity in place. They range in complexity, from specific tasks (*e.g.*, locating values) to higher level tasks (*e.g.*, data exploration). Depending on the *task*, the *perception* and *interaction* steps may be influenced (Jun et al., 2011).

Doleisch (2007) states that the goals of computational visualizations are threefold: exploration, analysis, and presentation. Although these three goals appear to follow a very linear sequence, this is not usually the case in real-world applications. Many users navigate data in an epistemic fashion, without

having any questions a priori and hoping to find new information through a knowledge discovery process (de Sousa & Barbosa, 2013). This is inherently exploratory and more inductive than traditional statistical methods (Mennis & Guo, 2009).

2.2

Why Narratives?

Data stories – presenting data and findings using visualizations in a narrative context – can be used to convey knowledge, share and interpret experiences (Elias et al., 2013). Using stories, data can become more interesting, memorable, comprehensible, credible, and accessible to the general public (Ma et al., 2012).

Moreover, storytelling can be used as an asynchronous two-way communication between authors and readers, in which the former can communicate their findings and insights, while the latter can ask questions directly on the story (Elias et al., 2013). Such an asynchronous environment can increase the scalability of group-oriented analysis (Heer & Agrawala, 2008). Compared to face-to-face collaboration, it can even result in higher-quality outcomes, since it results in broader discussion, complete reports, and more elaborate solutions (Benbunan-Fich et al., 2003).

The same dataset can present two different stories depending on how the narratives about data are constructed. Mennis & Guo (2009) stated that “the data cannot tell stories unless we formulate appropriate questions to ask and use appropriate methods to solicit the answers from the data.”

There are, however, important differences between traditional forms of storytelling and *data stories*. Wojtkowski & Wojtkowski (2002) argue that the main difference is the complexity of the content that is being told.

Segel & Heer (2010) point out another difference: the linearity and the control level of the narrative. The authors highlight that, whilst traditional storytelling narrates a set of events in a sequence controlled by the author, *data stories* usually offer some degree of interactivity, inviting verification, the posing of new questions, and the exploration of different paths and explanations.

Segel & Heer conclude that *data stories* exist somewhere along a spectrum of author-driven and reader-driven perspectives, summarized in table 2.1. When designing a *data story*, the central concern should be with balancing author-driven elements – the narrative structure – and reader-driven elements – the interactive exploration.

This spectrum seems in line with the taxonomy proposed by Wohlfart &

Table 2.1: Narrative visualization perspectives spectrum.

Characteristic	Author-driven	Reader-driven
Ordering	Linear ordering of scenes	No prescribed ordering
Messaging	Relies heavily	None
Interactivity	None	High (almost free)
Examples	Movies, comics, non-interactive slideshows	Visual analysis tools (such as Tableau or Spotfire)
Best when	Goal is storytelling or efficient communication	Data diagnostics, pattern discovery, and hypothesis formation

Adapted from Segel & Heer (2010)

Hauser (2007, *apud* Ma et al., 2012) regarding the author’s and reader’s degree of control:

Passive storytelling: Author has full control and reader does not interact with the story.

Storytelling with interactive approval: The narrative stops at certain specific points and the reader gets temporary control. After the interactive exploration, the narrative continues as originally intended.


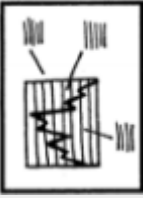
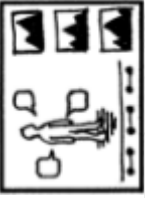
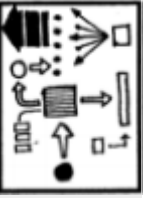


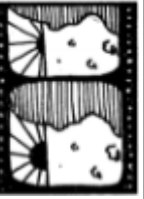
Semi-interactive storytelling: The reader can take control for an entire section of the story.

Total separation from the story: Readers can roam free, detaching from the narrative and engaging in exploration with total freedom.

Besides this author/reader-driven spectrum, narrative visualizations can have different *genres*. In their work, Segel & Heer (2010) studied some real-world examples of *data stories* and ended up with seven *genres* (summarized in table 2.2), varying mainly in the number of frames (distinct visual scenes, in time and/or space) and in the ordering of visual elements.

These *genres* are by no means mutually exclusive. They can also be combined with other features – such as interactivity and messaging – to fit in the desired place at the aforementioned author/reader-driven spectrum.

Table 2.2: Narrative visualization genres comparison.

Narrative visualization genre	Distinct visual scenes		Ordering
	Number of frames	Multiplexed in time Multiplexed in space	
 Magazine style	Single	n/a	Linear
 Annotated chart	Single	n/a	Directed
 Partitioned poster	Many	No	Loose
 Flow chart	Many	No	Directed
 Comic strip	Many	No	Linear
 Slide show	Many	Yes	Linear
 Video	Many	Yes	Linear

Adapted from Segel & Heer (2010)

3 BONNIE

There is a story in your data. But your tools don't know what that story is. That's where it takes you – the analyst or communicator of the information – to bring that story visually and contextually to life.

Cole Knaflic, *Storytelling with Data (2015)*

Given the importance of VAApps, there is surprisingly little support to communicate findings discovered in those applications (Elias et al., 2013; Bach et al., 2016; Knaflic, 2015). Users have to rely on their own ability to document the knowledge discovery process and generate different kinds of documents to disseminate the information. As Knaflic (2015, p 2) states: “being able to visualize data and tell stories with it is key to turning into *information* that can be used to drive better decision making.”

To bridge this gap, we developed BONNIE (Building Online Narratives from Noteworthy Interaction Events) (Segura & Barbosa, 2016). It is a framework to log, revisit, and explore user interaction history from a web VAApp (to simplify, we will refer to this VAApp as the *source system* or simply SrcSys from now on). From the user interaction history, the user is able to recreate the visualization from any given moment and generate a narrative containing those visualizations.

The idea behind our framework is to empower the user to revisit the sequence of steps he or she took in the SrcSys that led him or her to an insight. By choosing the desired steps, the user can create a narrative, containing the visualizations and textual annotations, to document and share the discovery.

As a user interaction history visualization framework, the communication with BONNIE actually begins with the SrcSys. Figure 3.1 depicts the full interaction sequence, considering the same user interacting with both systems.

The sequence starts with the user interacting with the SrcSys (1). The user's interaction with the SrcSys is logged (2) to be later presented in BONNIE (3). The same user who interacted with the SrcSys system now interacts with BONNIE (4), interpreting the visualization, revisiting the steps he or she took in the SrcSys and electing which ones will be part of the narrative.

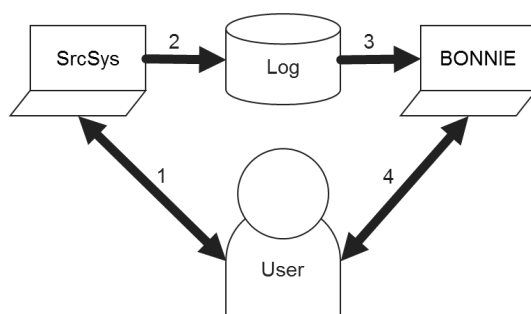


Figure 3.1: BONNIE interaction sequence.

Given this interaction sequence, our framework should support two distinct groups of users with very different goals: the SrcSys’s development team and BONNIE’s final user. Each group has their own set of requirements:

- SrcSys’s development team
 - Log interaction events so they can be visualized in BONNIE;
 - Describe the SrcSys visualizations so they can be used by BONNIE;
 - Save the visualizations states, so the visualizations for a given moment can be recreated.
- BONNIE’s final user
 - Present the user interaction history in a comprehensible manner;
 - Allow building a narrative from a subset of interaction events; and
 - Display the narrative as an interactive, although non-editable, web page.

To implement our vision, we used Node.js¹ runtime, developed with the TypeScript² language, and deployed it using IBM Bluemix.³ We considered the architecture illustrated in figure 3.2. We considered that a SrcSys is comprised mainly of a *web user interface*, which connects with a *data service interface* and displays data with *visualization components*. A *data service interface* may connect with a data service managed by the SrcSys team or a third-party to gather data. A *visualization component* contains one or more visualizations, which can be provided by an external *visualization library*. The SrcSys must register the interaction events in a *user history log* so we can provide the traceback to users.

BONNIE’s UI shares most components with the SrcSys, creating a visualization from the previously registered *user history log*. Moreover, it uses

¹<https://nodejs.org>

²<https://www.typescriptlang.org/>

³<http://www.ibm.com/cloud-computing/bluemix/>

the same *data service interfaces* and *visualization components* from SrcSys (and, consequently, the same *visualization library*) to recreate the SrcSys’s *visualization components*.

Figure 3.3 shows BONNIE’s UI. It has two main components: a *history visualization* and a *narrative builder*. They work closely together so the user can visualize the interaction history and choose the relevant steps to create the desired narrative.

For the remainder of this thesis, we will use WISE (Weather InSights Environment) (Oliveira et al., 2014; Ferreira et al., 2015) as our main SrcSys. All examples will be based on the WISE instrumentation, providing a “concrete” and real example of BONNIE in use. Moreover, all studies also used WISE as the SrcSys.

We chose WISE due to its exploratory nature to obtain insights. It has a fixed set of *visualization components*, coordinated amongst themselves. On the one hand, it was a good example of a VAApp, since it empowers the user to gain insights regarding weather data. On the other hand, it is comprised of a single page with few *visualization components*, allowing us to focus on developing our framework instead of dealing with the complexity of SrcSys itself. Moreover, we had full access to its source code, allowing us to instrument the code more freely.

WISE shows weather-related data, focusing on data from a given *forecast*. For our scenario, a new *forecast* is generated daily at midnight and each *forecast* comprises 48 hours of predicted data (*i.e.*, a forecast generated on January 1st at midnight predicts data up until January 3rd at midnight).

For the duration of a *forecast*, we have data for **every hour** (*i.e.*, the data *timestep* is of one hour) for many weather *properties* (*e.g.*, temperature, rain rate, humidity) for every *cell* (a latitude x longitude pair) of the *forecast grid*. When the user chooses a *configuration* (a *forecast*, a *property*, and a *timestep*), WISE displays the corresponding data for every cell on a map, using a categorical color scale.

Besides forecast data, WISE also shows observed data obtained from weather stations. The observed data shares the same categorical color scale,

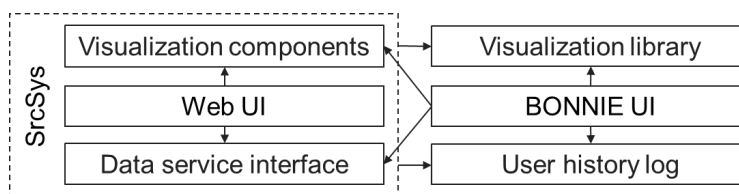


Figure 3.2: BONNIE’s framework considered architecture.

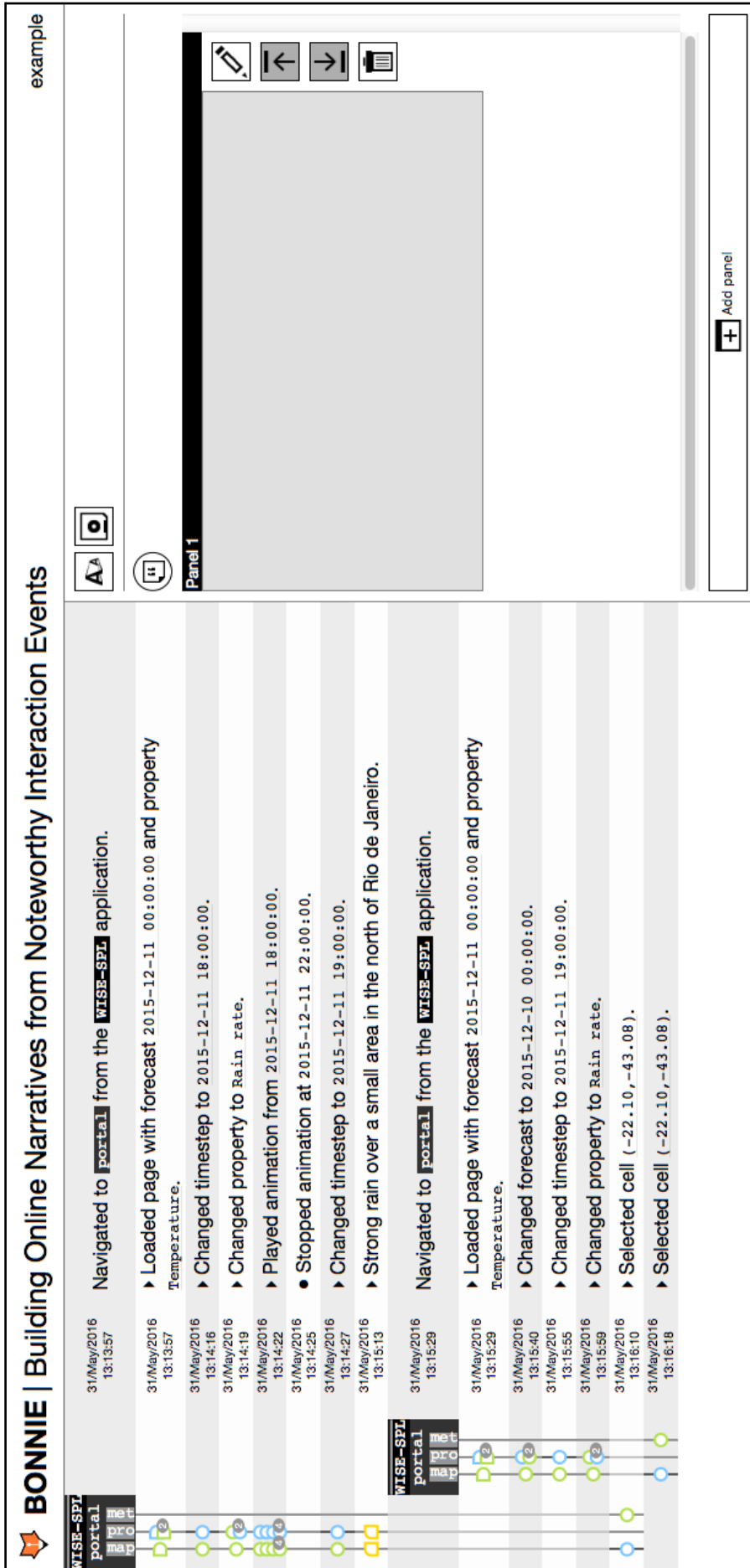


Figure 3.3: BONNIE's main UI.

but are represented using a different shape: forecast data is displayed using squares for each cell, whilst observed data is displayed using circles for each station. By showing observed data alongside forecast data, WISE allows not only the data exploration – detecting patterns and trends for forecast events – but also data verification and validation – comparing the forecast with observed data.

WISE’s main UI (henceforth referenced as *portal*) is shown in figure 3.4, which highlights its three main *visualization components*:

- a *map*, displaying forecast (the colored rectangles, each one associated to a *forecast cell*) and observed data (the colored circles, each one associated to a weather station);
- an *event profile*, a summary of the categorical distribution of the rain rate (classified as “No Rain”, “Weak Rain”, “Weak Moderate Rain”, “Moderate Rain”, “Strong Rain”, “Very Strong Rain”, or “Extremely Strong Rain”) through the duration of the forecast;
- *meteograms*, a series of line charts displaying the evolution of several weather properties over time for the selected cell in the map.

From our perspective, the *configuration* was **not** considered a *visualization component*, given that it resembles more a form (with its series of drop-downs UI elements) than a data visualization. Finally, the group of **BONNIE actions** is an overlay created by our framework which provides two functionalities. The left-hand side button allows the user to create a special kind of just-in-time annotation while interacting with the SrcSys (as it will be later explained). The right-hand side button shows when data is being logged and the amount of log data waiting to be sent to the BONNIE data service. When the user clicks on it, he or she is led to BONNIE, so as to easily check his or her previous interaction events.

In the next chapters, we discuss our log model used to register the *user history log* (chapter 4) and the required SrcSys instrumentation (chapter 5). Chapter 6 details our user history visualization notation whilst chapter 7 focuses on the support for building narratives.

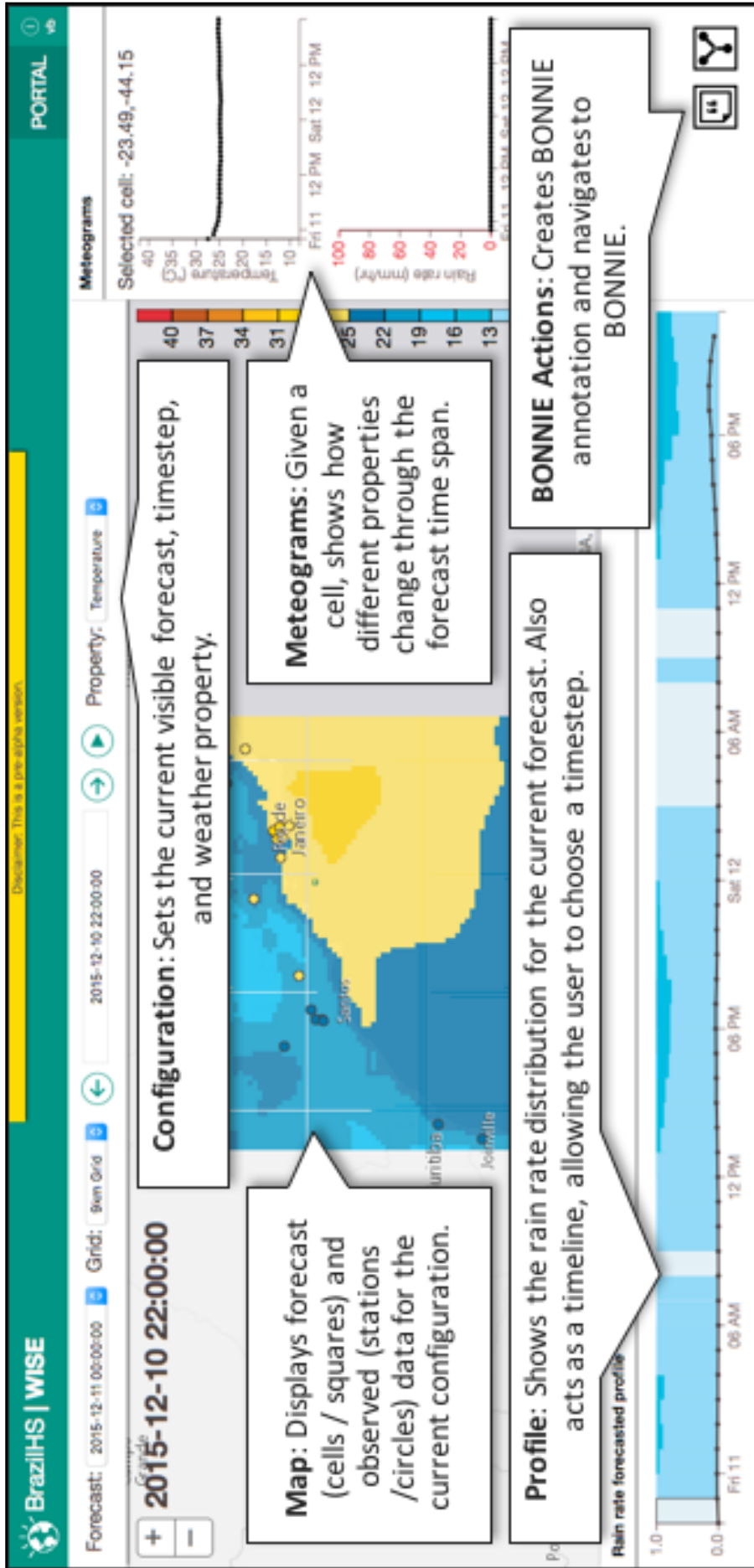


Figure 3.4: WISE’s main UI, highlighting its components.

4 Log Model

The *user history log* is the main communication mechanism between the SrcSys and BONNIE. Many logging approaches focus on low-level events which happen within a page (*e.g.*, mouse movement, mouse click, key presses) and consider only the HTML document object model. The resulting log data, therefore, lack a task-oriented semantics, focusing more on the operational level.

For example, the UsaProxy (Atterer et al., 2006; Atterer & Schmidt, 2007) solution creates a proxy, requiring little client and server side changes. It tracks, however, only events such as navigation between pages, mouse movements, and input events without giving any meaning to these actions.

Another possible approach is to compare the logged data with the structure of the web application task model (Paganelli & Paternò, 2002). It depends on mapping the task model beforehand and on keeping both constructs – log and task model – synchronized.

We propose a log model that integrates the logged data with a tactical level task model. It relies on the SrcSys’s developer to instrument the SrcSys code to express the task being performed. Instead of describing the operational level – the operation per se (*e.g.*, mouse click on a data point) –, the SrcSys’s developer should describe it at a tactical level – what is achieved with the operation (*e.g.*, highlight a data point). The strategic level – the user’s final goal (*e.g.*, compare different data points) – is not logged, since it depends on the user.

We developed the quasi-hierarchical model shown in figure 4.1 focused on the web VApp problem. Our model can be split in two parts: the ***definition hierarchy*** – the static definition of the SrcSys – and the ***interaction hierarchy*** – a record of the dynamic user interaction with the SrcSys.

The *definition hierarchy* reflects our interpretation of a VApp basic structure (as shown in figure 3.2). The ***source system definition*** (`SrcSysDef`) describes the VApp itself (*e.g.*, WISE). It has a collection of ***view definitions*** (`ViewDef`), the different pages (*e.g.*, WISE’s portal page) the user can go through (similar to the *web UI* in the architecture diagram). Each *view definition* has a collection of ***visualization component defini-***

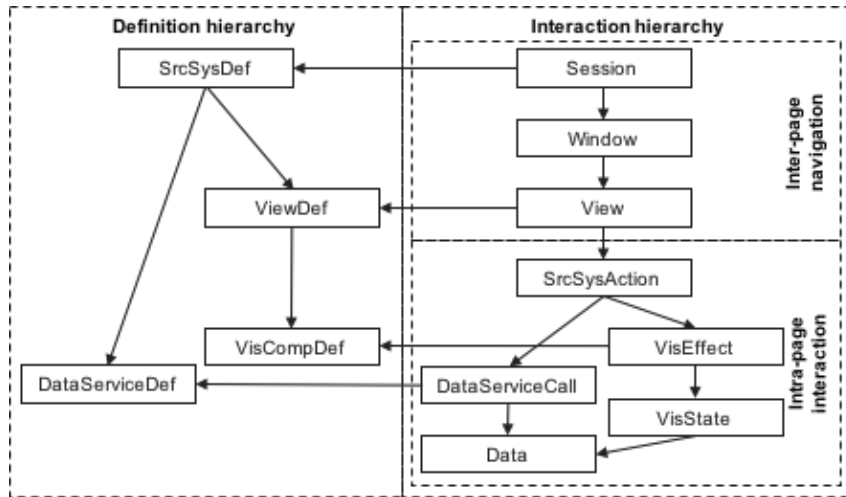


Figure 4.1: BONNIE's log model relations.

tions (`VisCompDef`). A *visualization component definition* represents a visualization available in the current *view*, which may be, for example, a chart, a map, or a group of charts (e.g., WISE's map, profile, and meteograms).

The *source system definition* also contains a collection of *data service definitions* (`DataServiceDef`). Contrary to the architecture figure (figure 3.2) – in which the *web UI* references a *data service interface* –, the *data service definition* is not a child of *view definition*, since multiple views can gather data from the same data service. Whilst the architecture diagram was created to illustrate the communication between components (a *web UI* uses a *data service interface*), the log model diagram illustrates the hierarchy (a *source system definition* has a collection of *view definitions* and another one of *data service definitions*).

The root of the *interaction hierarchy* is a *user session* (`Session`), created when the user logs into the SrcSys (thus referencing the *source system definition*). During a *session*, the user can open many browser windows and/or tabs – both are logged as `Window`. Inside a `Window`, the user may navigate through many web pages, corresponding to different *views* of the SrcSys (`View`, which references a `ViewDef`).

The interaction with a *view* can generate many *source system actions* (`SrcSysAction`). We are considering four different kinds of *source system actions*:

1. **User action:** action performed explicitly by the user interacting with the UI (e.g., when user clicks on a data point, selecting an element).
2. **Navigation action:** action performed when loading a *view* with the default parameters.

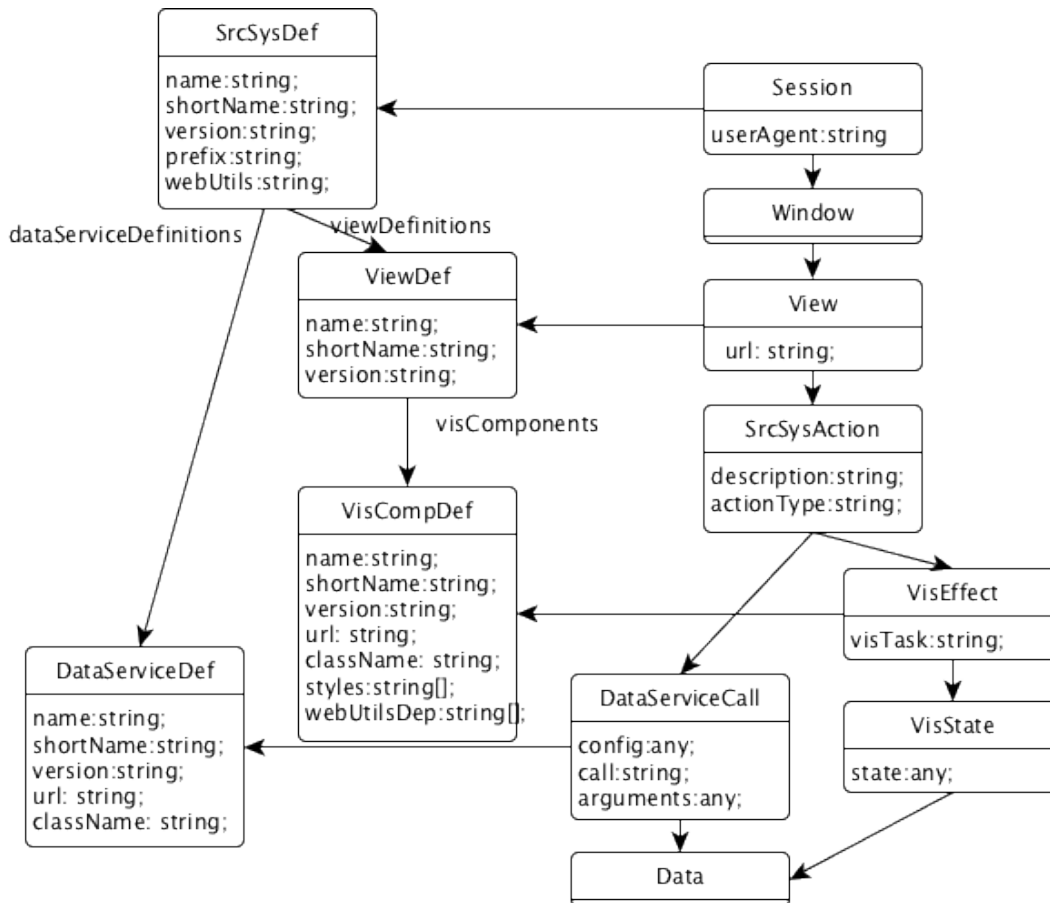


Figure 4.2: BONNIE's log model showing the attributes of each element.

3. **System action**: action performed automatically by the SrcSys (e.g., automatic refresh, gathering new data in the background).
4. **BONNIE annotation action**: action provided by BONNIE that allow users to create an annotation with the main purpose of supporting the history visualization and narrative creation in BONNIE.

Each *source system action* may trigger several **visualization effects** (`VisEffect`) and **data service calls** (`DataServiceCall`). A *data service call* uses a *data service definition* to gather **data** (`Data`). A *visualization effect* represents a *visualization component* change, referencing a *visualization component definition* and generating a new **visualization state** (`VisState`). Since data is encoded in the *visualization state*, a `VisState` may reference several `Data`.

Each element of the *log model* stores information according to figure 4.2. On the *definition hierarchy*, every element has at least three attributes: `version`, `shortName`, and `name`. These attributes are the element's main identifiers, allowing the SrcSys developer to update them accordingly. While the `name` can be expressed using any character, the `shortName` should be

limited to alphabetical characters. A combination of `shortName` and `version` is used to define the element's ID.

The `SrcSysDef` also has a `prefix` and `webUtils` attributes. The `prefix` is a short string used to create the id of all elements related to this SrcSys. The `webUtils` attribute is a URL to the *visualization library* (from the architecture in figure 3.2) used to create the visualizations in the SrcSys.

`VisCompDef` and `DataServiceDef` are elements that should be later instantiated by BONNIE. They have, therefore, a `url` and `className` attributes, indicating the URL to the JavaScript containing the class implementation and the name of the class, respectively. Additionally, the `VisCompDef` may contain an array of CSS URLs to style the visualization (`styles`) and an array of *visualization library* dependencies to be loaded dynamically (`webUtilsDep`).

On the *interaction hierarchy* side, the attributes vary according to the element. A `Session` has only the `userAgent`¹ attribute, describing the browser being used. A `View` has the `url` used during the original navigation. The `SrcSysAction` has an `actionType`, according to the aforementioned different *source system action* types. It also has a `description`, a message defined by the SrcSys developer describing the action at a tactical abstraction level.

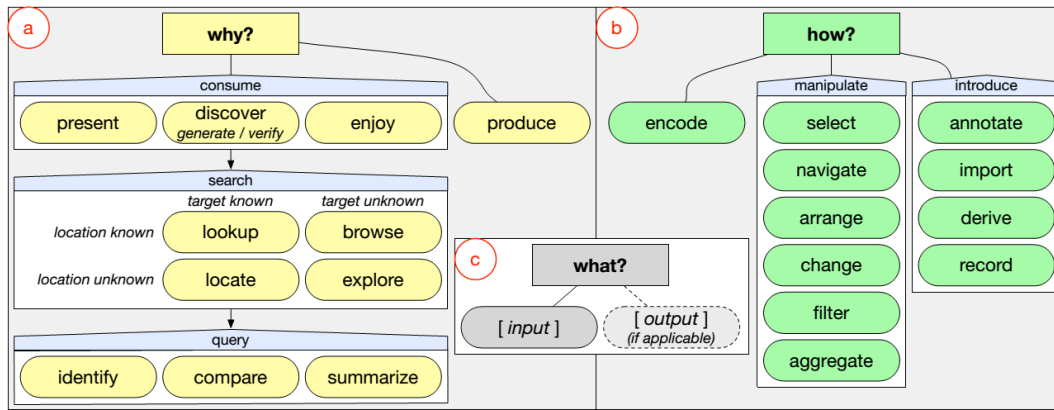
The `DataServiceCall` has information regarding how to gather data at run-time. The `config` is a JSON string describing the configuration of the `DataServiceDef` to instantiate it correctly. The same `DataServiceDef` could, therefore, be used in different scenarios (*e.g.*, gathering data from different compatible services). The `call` informs which was the data being requested and `arguments` is a JSON string with the arguments of that given call.

It is important to notice that, whilst we have an explicit `Data` element, it acts only as a placeholder. We are **not** storing the SrcSys data, given that this could lead to a replication of the whole database. Rather, we chose to make SrcSys *data service interface* calls at run-time, acknowledging the dependency that this approach entails.

The `VisEffect` has only a *visualization task* (`visTask`) attribute. Whilst *source system actions* are SrcSys specific, we envisioned *visualization effects* as independent from SrcSys. This allows the reuse of *visualization components* amongst different VAApps. Moreover, we chose to limit the variability of *visualization tasks* so we could display effects from different VAApps using the same notation.

For example, the same *visualization component* with a collection of line charts could be used in a traffic information VAApp to show the average speed of a street for a given time period and in WISE to show the meteograms.

¹<https://developer.mozilla.org/en-US/docs/Web/API/NavigatorID/userAgent>



Source: Brehmer & Munzner (2013)

Figure 4.3: Multi-level typology of abstract visualization tasks.

In this traffic information VApp, the user could notice a sudden decrease in average speed for a given day. Going to WISE, the user would verify that there was a strong rain event in that day, explaining the anomalous behavior. After this analysis, the user can go to BONNIE and check interaction events from both SrcSys (traffic VApp and WISE). The *visualization effects* would be displayed with a similar terminology for both interaction events, since it does not depend on the SrcSys. Since the *visualization tasks* are predefined, plotting data both in the traffic line charts and in the meteograms would be classified as the same *visualization task*, `Encode`.

On the one hand, the SrcSys developer provides a description, in free-form text, of *source system actions*, being encouraged to use terminology familiar to the SrcSys’s users. On the other hand, to each *visualization effect*, the SrcSys developer must associate a *visualization task* (`visTask`) from a predefined set.

The *visualization tasks* are based on a multi-level typology of abstract visualization tasks (Brehmer & Munzner, 2013), focusing only on the “how?” part of figure 4.3. In their work, Brehmer & Munzner studied about 50 previous works to come up with this typology, comparing their proposed terminology with the ones they found in the literature. They ended with eleven “leaf” nodes to describe interaction techniques, which we call *visualization tasks* in our work.

Table 4.1 provides a quick description of each *visualization task* and some similar terms from other terminologies. Due to the extent and coverage of Brehmer & Munzner’s work, we believe that this *visualization tasks* set can cover a wide variety of *visualization effects*.

Finally, the `VisState` may reference `Data` elements and has a single `state` attribute. The value of `state` is a JSON string that describes the `VisState`. The *visualization component* should be able to parse this `state`

Table 4.1: Descriptions of visualization tasks.

VisTask	Description	Similar terms
Encode	Codify data in the visual representation	create mapping, visualize, generate
Select	Demarcate one or more elements in the visualization, differentiating selected from unselected elements	brush, distinguish, emphasize, differentiate, highlight, mark, pick
Navigate	Alter user's viewpoint	focus, zoom, pan, rotate
Arrange	Organize visual elements	sort, rank, organize, permute, reorder, reconfigure
Change	Alter visual encoding	shift, scale, set, configure, distort
Filter	Adjust the exclusion and inclusion criteria for elements in the visualization	subset, exclude
Aggregate	Change the granularity of visualization elements	cluster, associate, simplify, merge
Annotate	Add graphical or textual annotations associated with one or more visualization elements	note, comment
Import	Add new elements to the visualization	add, create, generate
Derive	Compute new data elements given existing data elements	compute, calculate, estimate,
Record	Save or capture visualization elements as persistent artifacts	bookmark, history

Adapted from Brehmer & Munzner (2013)

and switch itself to that specific state.

We chose to have a separate `VisState` element (as opposed to having a `state` attribute in `VisEffect`) so we could reuse the same `VisState`, when possible. For example, if the user performs an undo operation, we would not need to create a new `VisState` element, but rather reuse the previous one. The same principle applies to our decision of having a placeholder `Data` element.

To store the log data, we are using the EW-PROV service.² This service expects the PROV-DM data model³ and the information to be serialized with the PROV-XML schema.⁴

We mapped our log model, therefore, to the schema shown in figure 4.4,

²Internal IBM project to store provenance data.

³<https://www.w3.org/TR/2013/REC-prov-dm-20130430/>

⁴<https://www.w3.org/TR/2013/NOTE-prov-xml-20130430/>

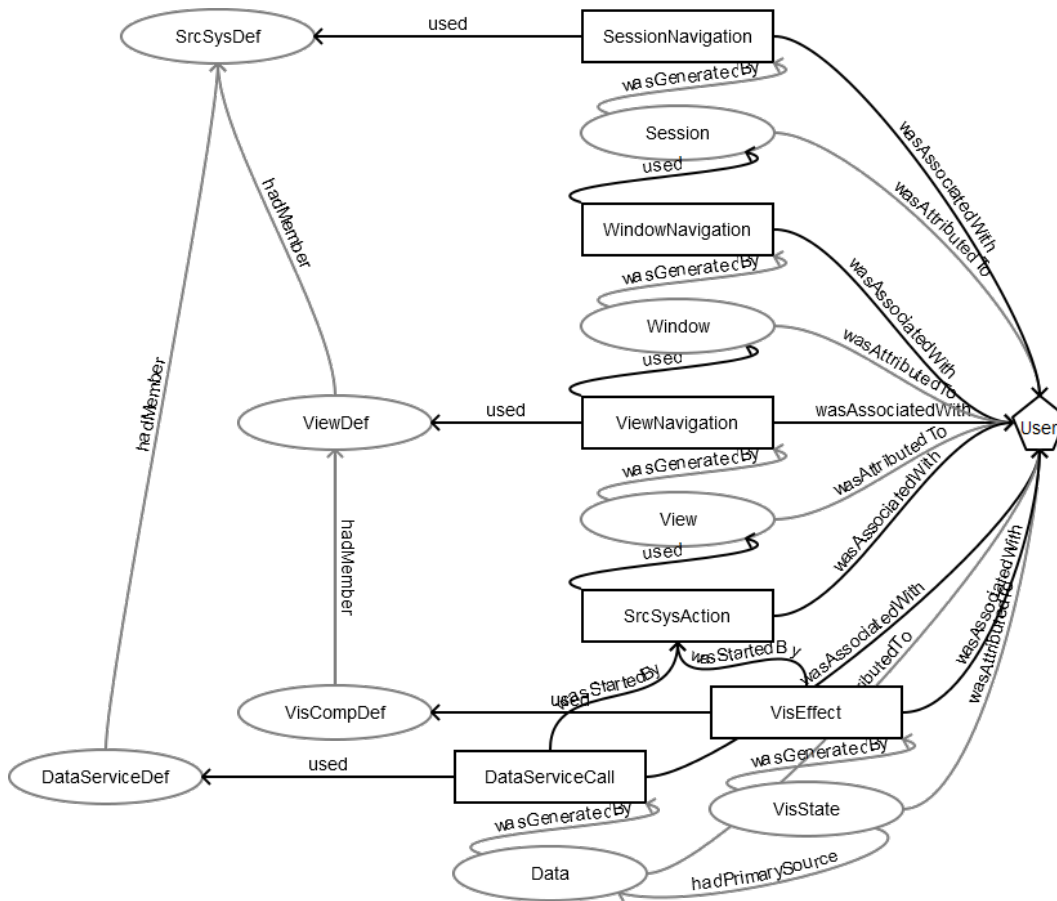


Figure 4.4: BONNIE’s log model as expressed using PROV-DM.

in which rectangles are *activities*, ellipses are *entities*, and pentagons are *agents*. Due to restrictions in the PROV-DM relation definitions, some new elements were introduced – namely `SessionNavigation`, `WindowNavigation`, and `ViewNavigation`. A complete example of the results of serializing each log model element can be seen in appendix B.

5 SrcSys Instrumentation

To comply with our framework, the SrcSys development team should follow some requirements, namely:

1. Keep definitions updated
2. Log interaction events
3. Use compatible *data services*
4. Use compatible *visualization components*

The next sections will focus on each requirement, explaining what the SrcSys development team should do to comply and how BONNIE framework may help with the tasks at hand.

5.1 Keep Definitions Updated

As previously discussed in chapter 4, our log model can be divided in two distinct hierarchies: the definition and the interaction hierarchy. The SrcSys development team is responsible for keeping them both synchronized. They must register the definitions and reference the registered IDs during interaction time, so the link can be set between the different hierarchies.

To aid the SrcSys developer, we made a simple web page in which the developer can post a definition hierarchy expressed as a JavaScript object (or in JSON format) and get the associated ID tree in JSON format. The TypeScript declaration of the input and output formats can be seen in listings 5.1 and 5.2, respectively.

Listing 5.1: Excerpt from the `ILogDefinitions` declaration file.

```
interface ILogStatic {
  shortName:string;
  name:string;
  version:string;
}

interface ILogSrcSysDef extends ILogStatic {
```

```

    prefix:string;
    viewDefinitions: {
      [devName:string]:ILogViewDef
    };
    dataServiceDefinitions: {
      [devName:string]:ILogDataServiceDef
    };
  }

interface ILogViewDef extends ILogStatic {
  visComponents: {
    [devName:string]:ILogVisComponentDef
  };
}

interface ILogVisComponentDef extends ILogStatic {
  url:string;
  className:string;
  styles:string[];
}

interface ILogDataServiceDef extends ILogStatic {
  url:string;
  className:string;
}

```

Listing 5.2: Returned IDs format from the definitions registration.

```

interface ILogSrcSysDefIDs extends ILogStatic {
  id:string;
  viewDefinitions: {
    [devName:string]: {
      id:string;
      visComponents: {
        [devName:string]: {
          id: string;
        }
      }
    }
  };
  dataServiceDefinitions: {
    [devName:string]: {
      id: string;
    }
  };
}

```


All definition inputs contain a `name`, a `shortName`, and a `version` (they all extend the `ILogStatic` interface). The `VisComponentDef` and `DataServiceDef` also contain the `url` to the corresponding JavaScript code and the `className` to instantiate the object. When registering definitions, the JSON output uses the same `devName` as in the input, so the developer can have some control over the structure of the generated ID tree. A concrete example of the input and output can be found in appendix C.

It is important to notice that, for the time being, the SrcSys development team is responsible for versioning. If non-breaking changes are made, the SrcSys development team may choose whether or not to change the `version` and/or the `url`. If breaking changes are made, it is strongly recommended to change the `version` and the `url`, as to generate a new ID. Moreover, it is up to the SrcSys development team to keep distinct older versions available, knowing that it may impact the visualization of older log information.

5.2 Log Interaction Events

Alongside the web page to register definitions, we developed helper code to aid the SrcSys development team to instrument their code to log the *interaction hierarchy*. We are exporting a global variable `BONNIE` that contains the `logger` object exposing the `ILogger` interface. The `logger` has methods to save log information according to the log model, as to be consumed by BONNIE later. The basic declaration can be seen in listing 5.3.

Listing 5.3: Excerpt from the `ILogger` declaration file.

```
interface ILogSrcSysDefId {
  prefix:string;
  srcSysDefId:string;
}

interface ILogViewDefId {
  viewDefId:string;
}

interface ILogDataServDefId {
  dataServDefId:string;
}

interface ILogVisCompDefId {
  visCompDefId:string;
  name:string;
}
```

```

interface ILogSrcSysActionId {
  srcSysActionId:string;
}

interface ILogDataId {
  dataId:string;
}

interface ILogDataIdsMap {
  [id:string]: ILogDataId
}

interface ILogger {
  startSession(user:ILogUser , sourceSystemId:ILogSrcSysDefId):
    void;

  logView(view:ILogView , viewDefId:ILogViewDefId):void;

  logSrcSysAction(srcSysAction:ILogSrcSysAction , sourceVisComp:
    IBaseVisComp<any>): ILogSrcSysActionId;

  logVisEffect(visEffect:ILogVisEffect , visState:ILogVisState ,
    srcSysActionId:ILogSrcSysActionId , visCompDefId:
    ILogVisCompDefId , dataIdsMap:ILogDataIdsMap):void;

  logDataServiceCall(dataServiceCall:ILogDataServiceCall ,
    srcSysActionId:ILogSrcSysActionId , dataServDefId:
    ILogDataServDefId): ILogDataId;
}

declare namespace BONNIE {
  export var logger:ILogger;
}

```

The SrcSys developers should call the `startSession` function just after the user logs into the SrcSys. They define the current *view* by calling the `logView` function. The logger will be responsible for saving the *user session*, *window*, and *view* IDs using a combination of JavaScript’s `sessionStorage` and `localStorage`.

The SrcSys developers must register the *source system action* by calling the `logSrcSysAction` function. One of the parameters of `ILogSrcSysAction` is the textual description of the *source system action*. The SrcSys developers can indicate a parameter in the description with the “[parameter]” markup (e.g., “Changed timestep to [[2015-12-11T19:00:00.000Z]].”). The `logSrcSysAction` function returns a `ILogSrcSysActionId` object, which should be used to log

triggered `VisEffects` and `DataServiceCalls`.

The other functions available in the `logger` (`logVisEffect` and `logDataServiceCall`) should only be called from *visualization components* and *data services*. To implement such elements, the developer should extend base abstract classes written in TypeScript, as explained in sections 5.3 and 5.4. This allows BONNIE to later consume the concrete classes to display the *visualization component* preview.

5.3 Use Compatible Data Services

The `BaseDataService` is responsible for logging *data service calls*. It takes a `DataServiceDef` ID in the constructor, so we can reference the correct *data service definition*. Listing 5.4 shows an excerpt from the `BaseDataService` file.

Listing 5.4: Excerpt from the `BaseDataService` file.

```
type CFunction<T> = (data:T, dataId:ILogDataId)=>void;

abstract class BaseDataService {
  constructor(dataServDefId:ILogDataServDefId);

  abstract getData<DataType>(callName:string, parameters:any,
    callback:CFunction<DataType>, srcSysActionId:
    ILogSrcSysActionId);

  protected logDataServiceCall(callName:string, parameters:any,
    srcSysActionId:ILogSrcSysActionId):ILogDataId;
}
```

The “concrete” implementation of a *data service interface* has two main concerns:

1. Call the `logDataServiceCall` method whenever data are fetched.
2. Implement the `getData` method, which receives the same parameters saved in the `logDataServiceCall` and an additional callback function that should be called when data are fetched.

5.4 Use Compatible Visualization Components

Similarly, the `BaseVisComp` takes a `VisComponentDef` ID in the constructor alongside the CSS selector of the `div` HTML element where the *visualization component* will be created. It is a generic class that receives a `VisStateType` type and keeps track of the current *visualization state* (with the `visState`

variable) and encoded data (with a data map). The data map associates an `id` as defined by the *visualization component* developer, and a `Data ID`, obtained after the `BONNIE.logger.logDataServiceCall` function. Listing 5.5 shows an excerpt from the `BaseVisComp` file.

Listing 5.5: Excerpt from the `BaseVisComp` file.

```
interface IDataMap {
  [id:string]:ILogDataId
}

abstract class BaseVisComp<VisStateType> implements
  IBaseVisComp<VisStateType> {
  protected visState:VisStateType;

  constructor(divCssSelector:string, visCompId:ILogVisCompId);

  abstract loadVisState(visState:VisStateType, dataMap:IDataMap
    ):void;

  protected updateDataMap(id:string, dataId:ILogDataId);

  logVisEffect(visTask:EVisTask, srcSysActionId:
    ILogSrcSysActionId):void;
}
```

The “concrete” implementation should, therefore, have four main concerns:

1. Update the current `visState` whenever needed.
2. Update the current data mapping whenever encoded data changes.
3. Call the `logVisEffect` function when there is a change in the *visualization component*.
4. Redraw itself from a saved `visState` and a `dataMap` whenever `loadVisState` is called.

6

User Interaction History Visualization

Many studies surveyed time-oriented data visualization (Silva & Catarci, 2000; Chung et al., 2005; Storey et al., 2005; Aigner et al., 2007b). Following the categorization schema proposed by Aigner et al. (2007b), we envisioned our visualization as involving:

– Time

- **Temporal primitives:** Time points (*vs.* time interval). Since we focus on interaction events, we are interested in instants in time, not extended periods of time.
- **Structure of time:** Linear (*vs.* cyclic and branching). Since we are considering the user’s interaction log, they are restricted to the linear passage of time. The user may explore different scenarios, with a lot of back-and-forth. This data exploration may branch into different scenarios, but it all happens in linear time.

– Data

- **Frame of reference:** Abstract (*vs.* spatial). Our data does not have an inherent spatial layout (*i.e.*, conditioned by natural circumstances or modeled realities). We could consider the *visualization components* layout in the SrcSys as a reference, but it is not an inherent characteristic – it can change depending on the SrcSys version or even the device used to access the SrcSys. Our visualization, therefore, should not be restricted by it.
- **Number of variables:** Multivariate (*vs.* univariate). We want to display as much information about the log model as possible. For example, for a given *visualization effect*, we want to encode information regarding the *visualization component* and the *visualization task*.
- **Level of abstraction:** Data abstraction (*vs.* data). Given the amount of data, we have to aggregate data in some way, exploring the hierarchical nature of our log model.

– Representation

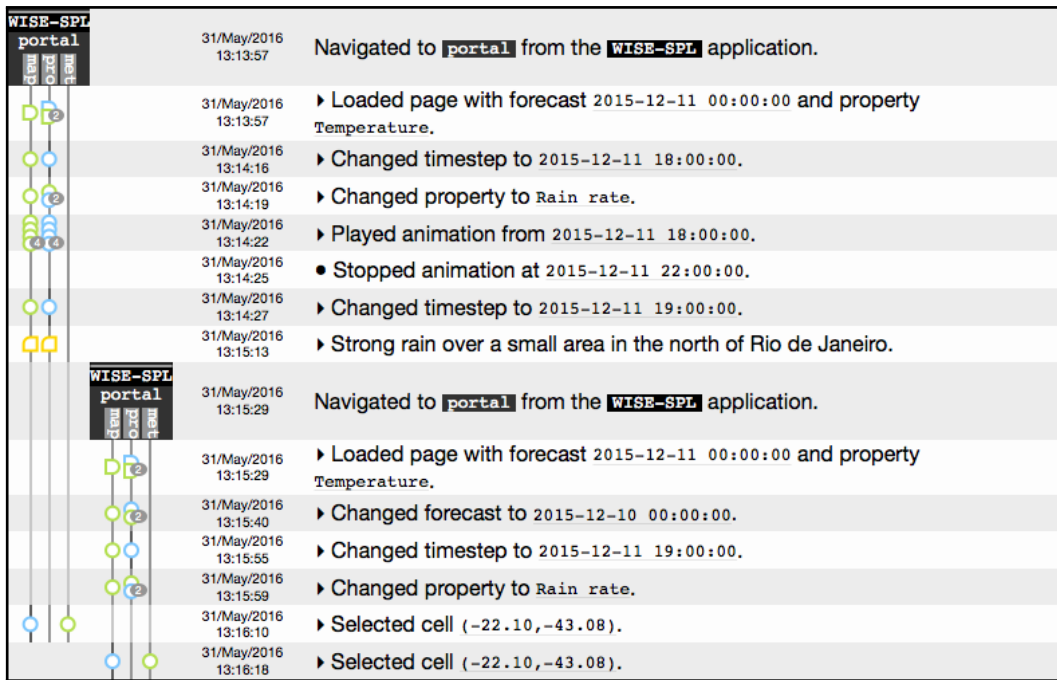


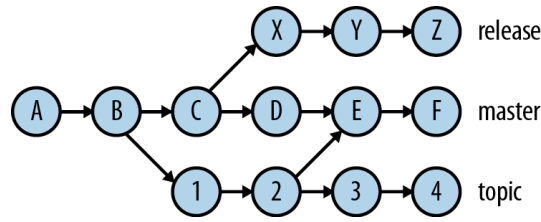
Figure 6.1: User interaction history visualization.

- **Time dependency:** Static (*vs.* dynamic). Although interactive, our visualization should not rely on the physical dimension of time to display information (*e.g.*, require to play an animation). We want to display the interaction history at a glance, allowing users to detect patterns or key navigation events. Moreover, the visualization already has the time dimension from when the logged interaction events took place. Adding another time dimension to display these events could bring an additional complexity.
- **Dimensionality:** 2D (*vs.* 3D). We believe that analysis in two dimensions is easier and effective, without introducing problems like projection and occlusion.

We propose the history visualization seen in figure 6.1. It shows the logged interaction events using textual descriptions combined with a graphical representation. We were inspired by version control systems’ commit graph, such as the GIT commit graph shown in figure 6.2. They are widely used, tackle a similar problem (time-based commit events), and handle branching issues.

Contrary to the commit graph, however, the interaction events in our visualization are organized from the oldest one to the most recent one. Reading the textual descriptions from top to bottom would, therefore, follow the events’ chronological order, making it easier to understand the history.

Another difference is that we do not handle branching in our visualiza-



Source: <http://chimera.labs.oreilly.com/books/1230000000561/ch01.html#fig0101>

Figure 6.2: Example of GIT commit graph.

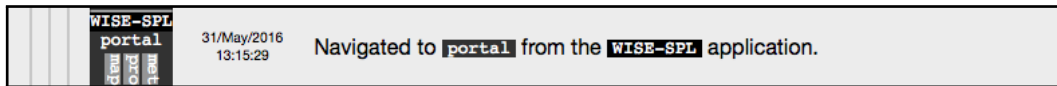


Figure 6.3: Example of a navigation row.

tion. A page navigation creates a new flow, with a new set of vertical lines. Each vertical line is a *visualization component*, and the *visualization effects* are represented as symbols over each line. It is a similar approach to the work proposed by Yoon et al. (2013), in which they represented each file of a version control repository as a line and marked revisions over the lines.

Our history visualization has three different kinds of rows: navigation, action, and effect rows. The rows background color are just for legibility, alternating between independent rows.

A *navigation row* (shown in figure 6.3) is related to a page navigation, *i.e.*, with a `View` log model element. Its representation is characterized by the break in the graphical representation flow. The break shows the *source system* (`SrcSysDef`), the *view* (`ViewDef`), and the *view's visualization components* (`VisCompDef`).

From each *visualization component* name emerges a vertical line which represents how long that given *view* was active (*i.e.*, there may still be actions happening in that *view*). For example, figure 6.3 shows three different visualizations in that given view: map (on the left), profile (“pro”, in the middle), and meteograms (“met”, on the right).

An *action row* (shown in figure 6.4) represents a `SrcSysAction` log model element. It shows the description defined by the SrcSys development team to describe the action performed in SrcSys. As previously mentioned, it uses a vocabulary specific to each *source system* and highlights the parameters of each action by formatting them in monospaced font with dotted underline.

A *source system action* may have caused effects on *visualization components* (*i.e.*, generated `VisEffect`s). These effects can be seen as nodes in the representation. A single action may trigger many effects and each node position indicates which *visualization component* was affected. For example, in figure 6.5(a), the top *action row* has caused 8 effects in total – 4 in the

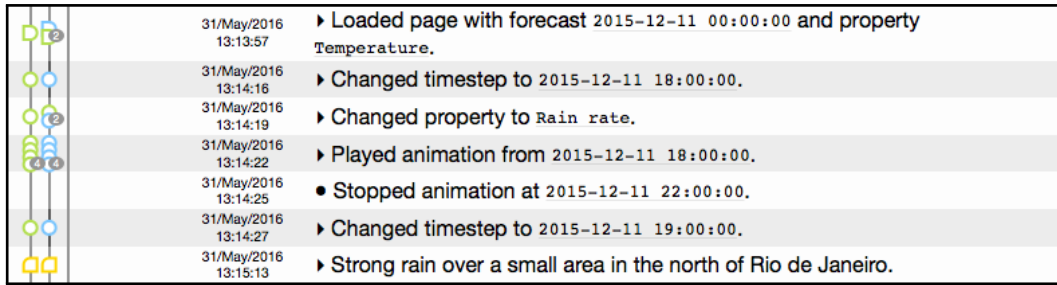
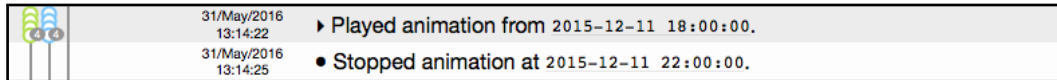
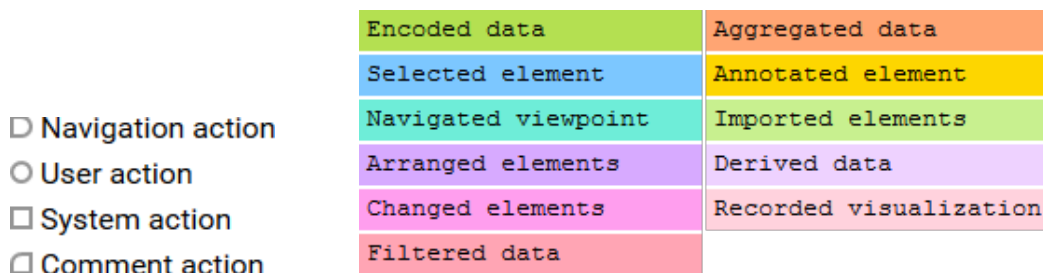


Figure 6.4: Examples of action rows.



6.5(a): Effects shown while the source system action rows are collapsed.



- ◻ Navigation action
- User action
- ◻ System action
- ◻ Comment action

6.5(b): Node shapes.

6.5(c): Node colors.

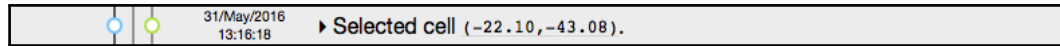
Figure 6.5: Nodes representing visualization effects.

left-hand side *visualization component* (map) and 4 others in the middle one (profile) – while the bottom one had no effect on any *visualization component* – they kept the same *visualization states*. The node shapes represent the different `SrcSysAction` types (figure 6.5(b)) and the colors represent the different *visualization tasks* (figure 6.5(c)).

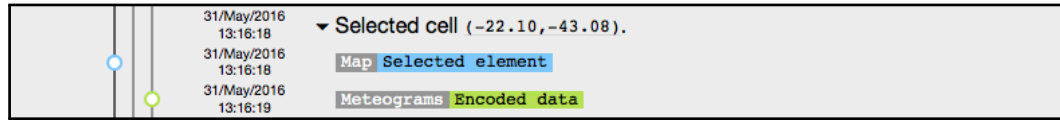
Finally, *effects rows* can be seen when expanding an *action row* (all rows start collapsed by default). When clicking on an *action row* with *visualization effects*, it goes from collapsed (indicated by the “▶” symbol) to expanded (indicated by the “▼” symbol), as shown in figure 6.6. The *effect rows* become visible, with the same background as its parent *action row*, and the *effect nodes* “slide down” to their respective rows. An *action row* with a “●” symbol indicates that it does not have any *visualization effect*.

Effect rows are characterized by the colored tags in the description, sharing the same color code with the corresponding *Effect node* (figure 6.6(b)). Contrary to the *action row*, *effect rows* are domain-independent and categorized according to the *visualization tasks*.

The *visualization component lines* can also hint in which *visualization component* the action occurred (darker segments) and the unrelated *views* (lighter segments). For example, in figure 6.7, we can notice that the first



6.6(a): Collapsed



6.6(b): Expanded

Figure 6.6: Example of effect row.

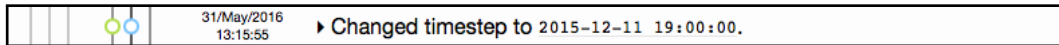


Figure 6.7: Line colors detail.

3 lines are lighter, indicating that this *view* is unrelated to the action. The 4th and 6th lines are in their neutral color, whilst the 5th line is in a darker tone. This means that the visualization was related to the action (in this case, the action was changing the timestep by clicking on the profile visualization component).

When hovering over an *action row*, a comment button appears at its rightmost edge (figure 6.8(a)). Clicking on the comment button, a comment pane appears (figure 6.8(b)), allowing the user to add comments to the corresponding action. If an *action row* has comments, the comment button is always visible, showing the number of comments associated to that action (figure 6.8(c)).

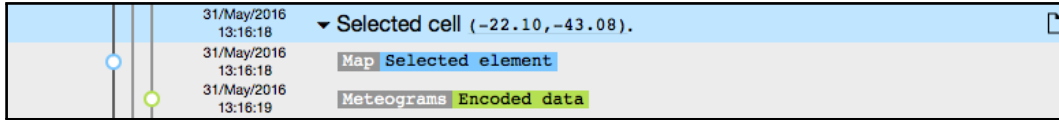
When hovering over an *effect row* (figure 6.9(a)), an information button appears to show the metadata associated with that impact. In the popup dialog (figure 6.9(b)), it is possible to see the underlying metadata code regarding the encoded data, the description of the new *visualization state*, and a preview of the visualization.

The described graphical representation was the result of several iterations regarding the notation. In the next sections, we will present two studies performed with the history visualization in different versions of the notation. Section 6.1 describes an analytical evaluation using Physics of Notation (PoN) (Moody, 2009) and Cognitive Dimensions of Notation (CDN) (Green & Petre, 1996). Section 6.2 presents a user study analyzing only the history visualization.

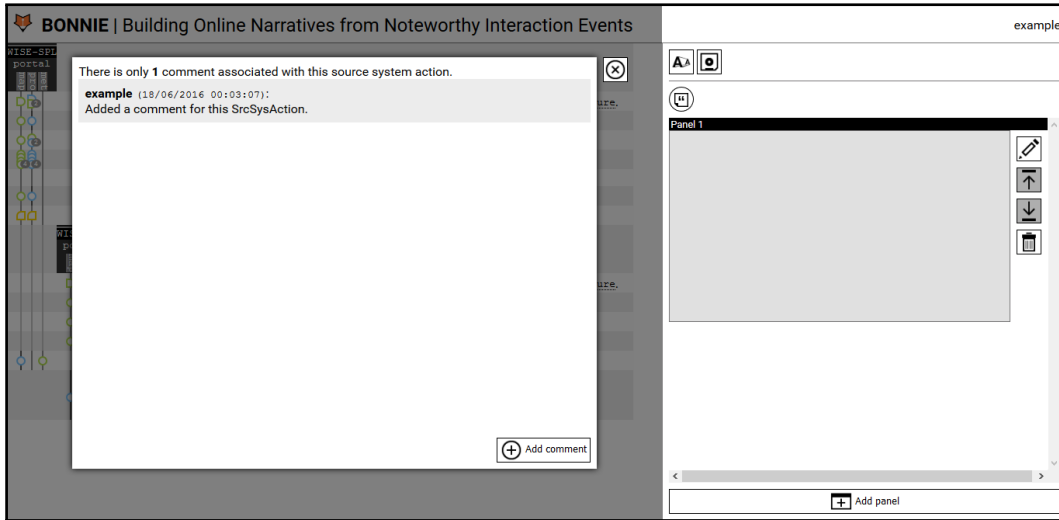
6.1

History Visualization Analytical Study with PoN and CDN

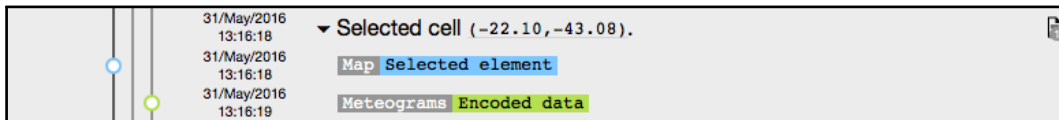
Before investing time and other resources in conducting user evaluations, we applied some analytical approaches (Segura et al., 2016) to (i) identify po-



6.8(a): Hovering over an action row.

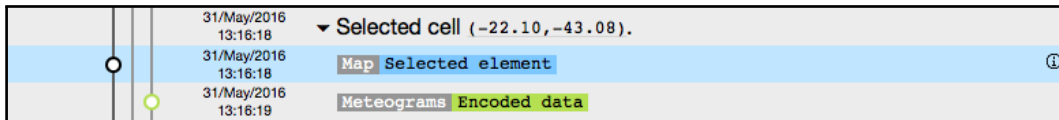


6.8(b): Action row comment pane.

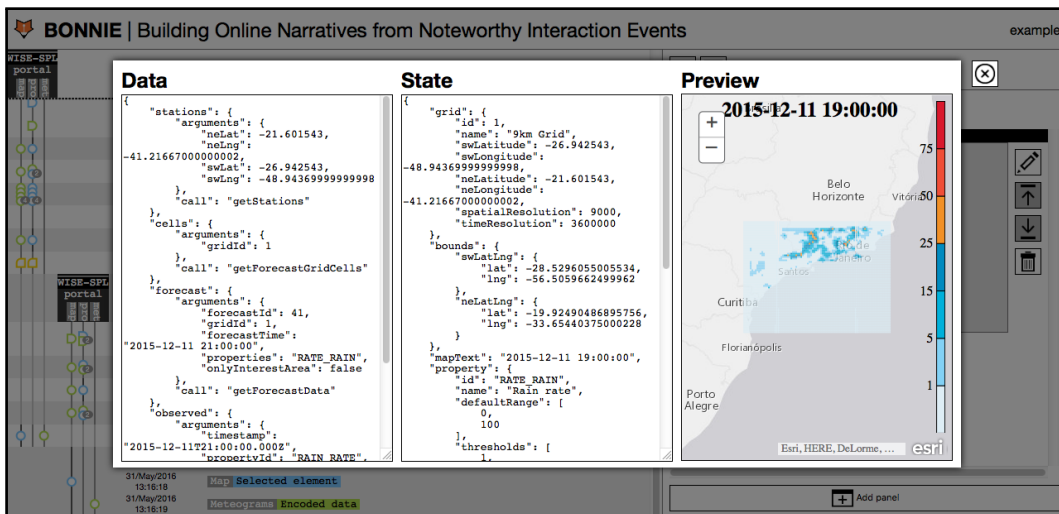


6.8(c): Action row with comment.

Figure 6.8: Action row comment feature.



6.9(a): Hovering over an effect row.



6.9(b): Effect row developer metadata.

Figure 6.9: Effect row metadata information feature.

tential usability problems, (ii) guide (re)design, and (iii) establish a common ground to compare and discuss design alternatives. Since the visual representation has a profound effect on the usability and effectiveness of the user interface, we decided to focus on analyzing the visual elements first.

We selected the Physics of Notation (PoN) (Moody, 2009) and the Cognitive Dimensions of Notation (CDN) framework (Green & Petre, 1996) to assess our early implementation, because they both handle visual representation in a complementary way. On the one hand, PoN focuses on syntax and physical properties of the visual notation, supporting detailed, symbol-by-symbol analysis. On the other hand, CDN considers the visual notation in a context of use – the notation semantics –, supporting a “broad brush” analysis. Both approaches provide a vocabulary to support design discussions, very common in early stages of design.

PoN provides “a set of principles for designing cognitively effective visual notations” (Moody, 2009). This set was compiled from theoretical and empirical evidence, establishing a scientific basis to compare, evaluate, improve, and construct visual notations. PoN has been used to verify the cognitive effectiveness of visual notation by evaluating the syntax of those notations (Moody & Hillegersberg, 2009; Moody et al., 2010; Genon et al., 2011).

The CDN framework defines a set of design principles – the cognitive dimensions (CDs) – for creating or evaluating notations, user interfaces, and programming languages used with information artifacts (Blackwell & Green, 2003; Green & Petre, 1996). CDN considers, evaluates, and discusses the notations and interaction languages regarding how well they support the intended activities.

CDN is a flexible analysis tool which deals with cognitive artifacts, not only visual notations (Green & Petre, 1996; Ferreira et al., 2012) and not necessarily with computational support (Petre, 2013). It is a “broad brush” analysis that provides a vocabulary to discuss various cognitive aspects of any notational system. PoN can be considered complementary to CDN, providing the type of detailed, domain-specific analysis that the authors of the CDN argued was necessary to supplement the analysis provided by CDN (Green & Petre, 1996; Petre, 2013).

Appendix D details both evaluations, discussing the analysis of each PoN principle and the issues raised by the CDN framework. From this study, several changes were implemented and incorporated into the current version of the notation:

- **New *navigation row***: Instead of a separate **Session** and **View** rows,

we merged them into a single navigation one, in order to: (i) save space, since different `View`s could share the same column, even if they are from different `Session`s; (ii) have an integrated legend for the `VisCompDef` vertical lines; and (iii) provide a more detailed context to the flow of `SrcSysAction`s, since the whole `SrcSysDef / ViewDef / VisCompDef` is represented in a single row.

- **“Colored tag” approach to the *navigation row*:** To better differentiate the descriptions that are specific to the SrcSys domain, we are using the “colored tags” both in the *navigation row* and in the *effect row*. The *action row* – the only one that is defined by the SrcSys developers – became the only row without “colored tags”.
- **Different node grouping:** Instead of a single node with a count badge, we now display all the nodes distributed vertically. This allows the user to have an overview of the different *visualization tasks* involved in the `SrcSysAction`. Moreover, it helps the user to perceived the amount of *visualization effects*, as it is dual-coded with the count badge.
- **Lighter *visualization component* line for unrelated *views*:** In a given *action row*, only the *visualization component* lines from the related *view* have the default gray shade: all the other lines have a lighter shade. This change aimed to help users perceived which *view* was related to each `SrcSysAction`.

6.2 History Visualization User Study

After the analytical evaluation and implementing the aforementioned changes, we conducted an empirical study. We started this study considering the interaction sequence depicted in figure 3.1. As we planned the study, we have decided to map the different players involved in this interaction sequence. Moreover, we were considering different points where miscommunication or misunderstanding could happen.

In the end, the interaction sequence diagram evolved into the one shown in figure 6.10. The sequence starts with the user planning actions to be performed in the SrcSys (1) and executing them while interacting with the SrcSys UI (2.a and 2.b). The user’s interaction with the SrcSys is logged to be later presented in the BONNIE UI. That log is expressed in a vocabulary extracted from the domain during the SrcSys development, by its designer (3.a, 3.b, and 3.c). This communication of the SrcSys designer about the user’s interaction is saved using the predefined log model (4), explained in section 4.

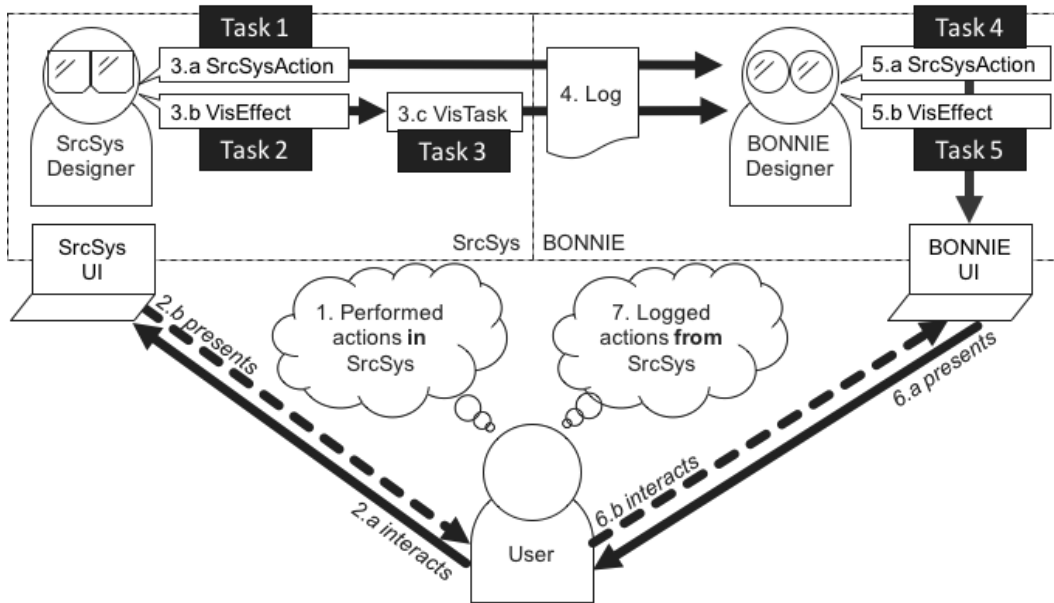


Figure 6.10: BONNIE interaction sequence considering the different players involved and their communication.

BONNIE’s designer expresses this log information through the BONNIE UI (5.a and 5.b), using the notation presented in section 6. The same user who interacted with the SrcSys system now interacts with the BONNIE UI (6.a and 6.b), interpreting the history visualization, revisiting the steps he or she took in the SrcSys, and creating a new mental model regarding the logged actions (7).

We developed the study tasks considering these different steps. We conceived three tasks (tasks 1, 2, and 3) to compare the participants’ mental model with the SrcSys designer’s expression of actions, whilst contextualizing the SrcSys terminology and the underlying BONNIE concepts. Two other tasks (tasks 4 and 5) were conceived to evaluate whether the participants were able to relate their mental models from two different moments: the one from interacting with the SrcSys and the other from interacting with the log represented in BONNIE.

We conducted a pilot study but, as the study materials and procedures did not change significantly (only a question was added to the final questionnaire), the pilot participant was considered in the analysis (denoted by P0). We then performed the study with four other participants (P1 to P4). All participants work with computer science and have different education degrees: Master’s and PhD. Appendix E details the study, explaining its procedure, documenting the materials used in the study, and presenting the collected results.

The study encouraged us to give more details about a *visualization effect*.

As P4 stated, we could use some sort of “metadata” to add information to the *visualization effect*. This led to the implementation of the *effect row* information pane, shown in figure 6.9(b).

The study ended up highlighting how the interaction with BONNIE is strongly affected by the decisions made by the SrcSys designer. Participants could not fully grasp the context of the history visualization given the choice of words and parameters defined by the SrcSys designer. For example, one participant could not find the desired cell because he was expecting to see the latitude and longitude as “(lat,lng)” but only found “latXlng”.

Given the results, we are planning to develop a set of guidelines to improve the communicability (de Souza, 2005) of BONNIE by instructing the SrcSys designer. For example, one guideline that emerged from the study would be to “always tell explicitly the result of the changes to parameters”, so the “changed to previous timestep” *user action* could be rewritten as “changed to previous timestep ([new timestep])”. Another guideline would be to “use the domain terminology”, which is akin to Nielsen’s guideline “match between system and the real world” (Nielsen, 1993). So, instead of representing the latitude/longitude pair as latXlng, it should be represented as (lat,lng).

To make the *visualization effect* metadata more useful to end users, the SrcSys designer should make the *visualization states* and *data service calls* human readable. For example, analyzing the stored log data, we saw that the parameters of the *data service calls* are being stored as they are consumed by the API, using some sort of database ID. For example, the call for a `getForecast` stores the parameter `id=41`, which will not mean anything to the end user.

The study validated our history visualization approach. The results were very promising, although the participants made several mistakes. The participants pointed out not having interacted with the SrcSys as the main reason for the mistakes, followed by not relating to the test scenario and not being used to the visualization. However, they were able to notice how the visualization could help them when viewing the user interaction history. Moreover, the feedback they provided was very valuable, leading to the implementation of new features (such as displaying the metadata information from *visualization effects*).

7

Narrative Builder

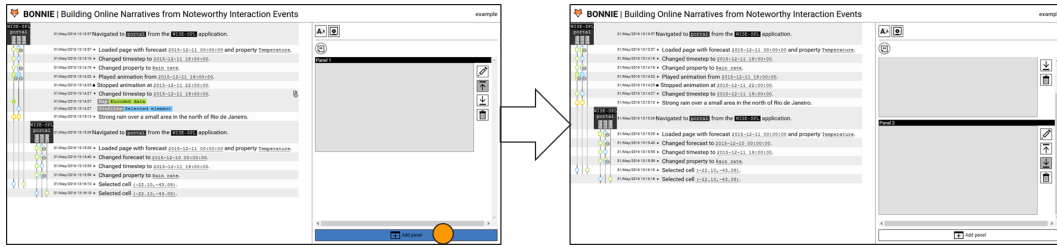
The narrative builder was created considering a slideshow / comics layout. Comics integrate images and text to communicate with an expressive and flexible language (McCloud, 1993). They can take small spaces to communicate complex information efficiently and effectively when compared to text-only (Green & Myers, 2010), either in print or digital displays (Bach et al., 2016).

In BONNIE, the narrative is built using three main components: *panels*, *textual elements*, and *visualization elements*. *Panels* structure the narrative, creating a sequence which the reader can go through in his or her own pace. When the narrative is displayed, each *panel* occupies the whole available screen space, so the reader must scroll through different *panels* to read the whole narrative in a linear fashion. *Textual elements* contain text defined by the narrative author. A *visualization element* corresponds to a `VisState` in the log model, representing a given *visualization component* at a given time of the interaction.

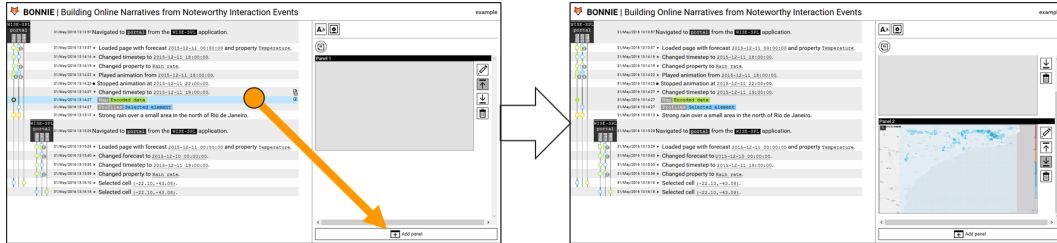
The main interaction method to build the narrative is drag-and-drop. *Panels* can be created by clicking on the “Add Panel” button (figure 7.1(a)) or by dragging an *element* to the button (figure 7.1(b)) or to an empty space in the narrative builder (figure 7.1(c)). In the latter two cases (when dragging an *element*), the new *panel* is created to contain the *element*.

Textual elements can be added by dragging an *action row* (figure 7.2(a)), an *action row* comment (figure 7.2(b)), or the *text element* icon (figure 7.2(c)) to a panel. When dragging an *action row* or an *action row* comment, the added *textual element* starts with a default text (the *action row* description and the comment itself, respectively). When dragging the icon, a pop-up appears so the user can edit the contents of the *textual element*. Once added, *textual elements* can be edited (to change their content) or deleted. *Textual elements* accept any valid HTML code, so the user is free to format the text as desired.

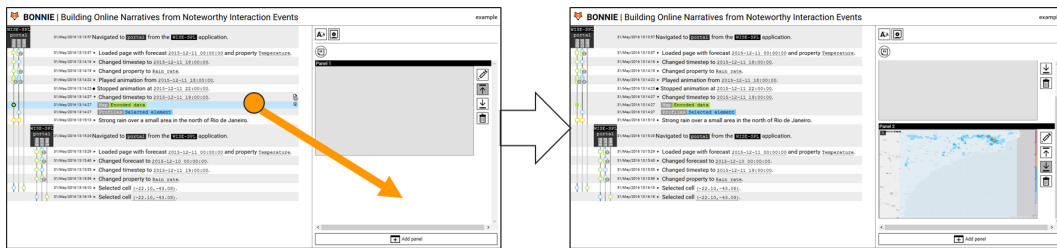
Similarly, *visualization elements* can be added by dragging an *effect row* (figure 7.3(a)) or an *effect node* (figure 7.3(b)) to a panel. Once added, *visualization elements* can only be deleted. Since they cannot be edited, we take advantage of the link between *visualization elements* and their corresponding



7.1(a): By clicking on the “Add Panel” button.



7.1(b): By dragging an element to the “Add Panel” button.



7.1(c): By dragging an element to an empty space.

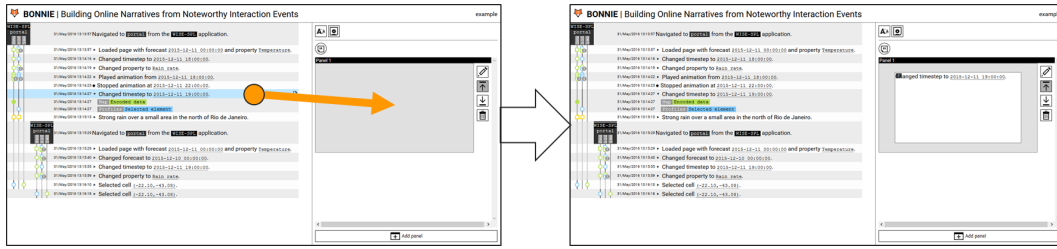
Figure 7.1: Adding a panel to the narrative.

visualization effects. To indicate that an *effect node* is being used in the narrative, the node changes its background from white to the *visualization task* color. When the user hovers over a *visualization element* in the narrative, the corresponding node/row is highlighted in the history visualization and vice-versa.

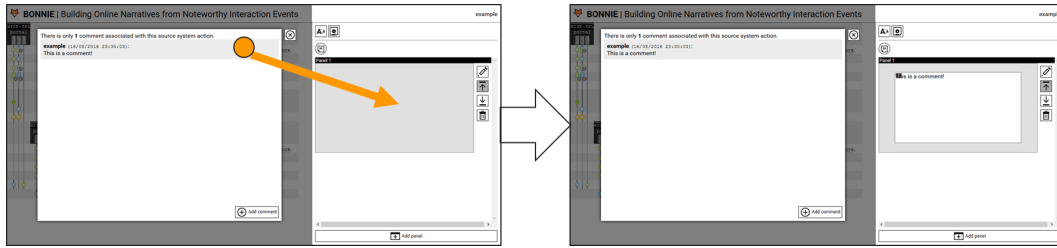
Multiple *textual* and *visualization elements* can be added to the same *panel*. Figure 7.4(a) shows the default layout mechanism: *visualization elements* are vertically distributed through the whole *panel* background, whilst *textual elements* cascade in the center area of the *panel* foreground.

On the *panels*’ right-hand sidebar, the user can find the “Advanced layout” (the pencil icon), reorder (the up and down arrows), and delete (the trashcan icon) buttons. The “Advanced layout” button opens a table showing every *element* in a row, with an identification number (the same one shown at the top-left corner of the *element*) and its top, left, width, height, and z-index values. In the future, we plan to allow direct manipulation of the *elements* (the “basic layout” option) to move and resize them.

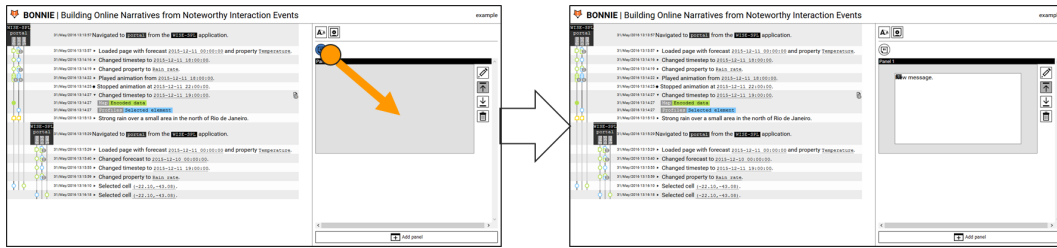
After the user creates a narrative, defining the *panels* and *elements*, he



7.2(a): By dragging an action row to a panel.



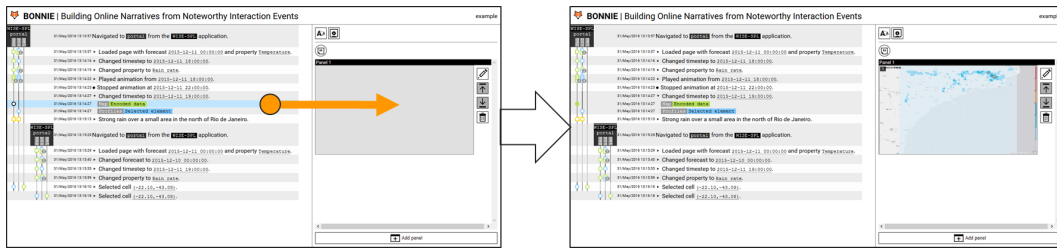
7.2(b): By dragging an action row comment to a panel.



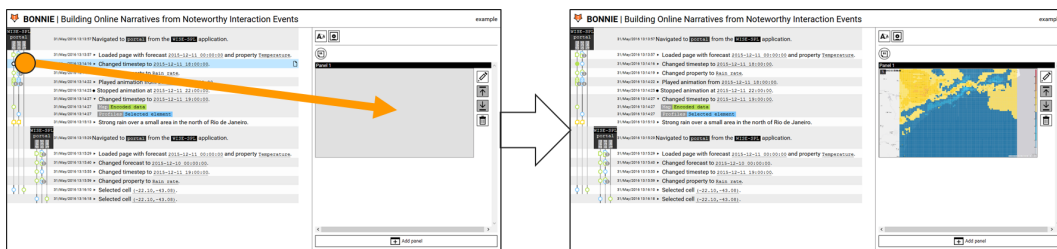
7.2(c): By dragging the textual element icon to a panel.

Figure 7.2: Adding a textual element to a panel.

PUC-Rio - Certificação Digital N° 1212408/CA

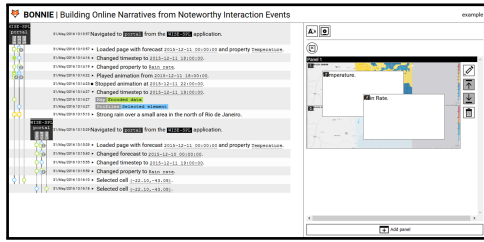


7.3(a): By dragging an effect row to a panel.

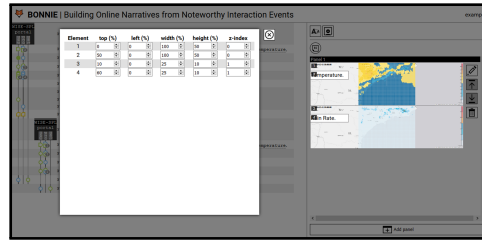


7.3(b): By dragging an effect node to a panel.

Figure 7.3: Adding a visualization element to a panel.



7.4(a): Default layout.



7.4(b): Edited layout.

Figure 7.4: Layout of textual and visualization elements.

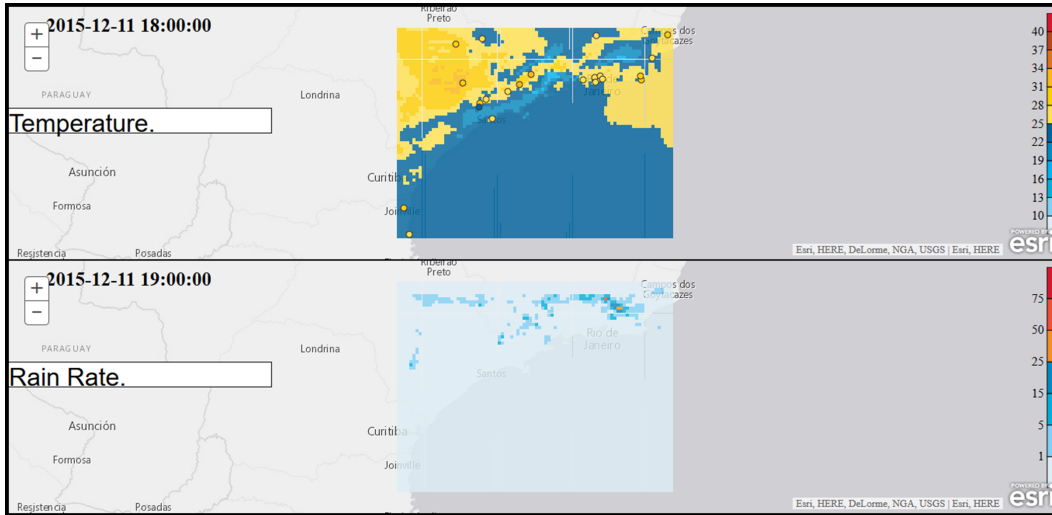


Figure 7.5: A single panel narrative.

or she can save the narrative. This creates a link to a web page in which the reader can view and interact with the narrative. Figure 7.5 shows a single-panel narrative.

7.1 Narrative Builder User Study

Following the previous history visualization study (discussed in section 6.2), we decided to apply another study with the same participants about three months later. This time, the study had only two tasks. Task 1 was similar to task 5 from the previous study, asking participants to find a given **VisEffect** given a video with a recorded WISE interaction. Task 2 asked participants to interact with WISE and create a narrative based on their analyses of WISE data.

After each task, the participants answered two questionnaires. The first questionnaire was very similar to the one from the previous study, focusing only on the history visualization part. The second one followed the original technology acceptance model (TAM) (Davis, 1989) and some constructs from TAM2 (Venkatesh & Davis, 2000), namely “intention to use”, “job relevance”,

“output quality”, and “result demonstrability”. A more detailed explanation of the study, all the used material, and the collected data can be found in appendix F.

The idea of answering the questionnaires after each task was to evaluate whether the actual interaction with the SrcSys would somehow impact the interaction with BONNIE. To avoid the learning effect of performing task 2 after task 1, we adopted a within-group (paired samples) design. We randomized the tasks order (P3 and P4 performed task 2 before task 1) so the learning effect of a user would be offset by another one. Consequently, the entire data set is not significantly biased by the learning effect (Lazar et al., 2010, p. 52).

Compared to the previous study, task 1 had fewer errors – 12 errors in the previous study (table E.8) and 6 in this one (table F.2). It is not clear, though, whether this improvement was due to changes in the history visualization representation or to the time gap between watching the video and actually performing the task. In the previous study, the participants could watch the video during the first three tasks, but not in the last two. In this study, participants watched the video just before performing the tasks.

Nonetheless, the addition of the “Stopped animation at [[timestep]]” row greatly improved the participants’ performance in finding the correct timestep for the map in the 2015-12-11 forecast. To find the correct timestep for the profile in the same forecast, P1 was able to use the animation action (with many *effect rows*) instead of using the more direct “Changed timestep to [[timestep]]” row (with only two *effect rows*). This was an indication that BONNIE provides multiple ways for users to achieve their goals, which we consider positive, especially when dealing with exploratory data analysis.

During the execution of the second task (free interaction with WISE and BONNIE), the participants were able to identify their interaction events sequence from the history visualization. They had, however, more difficulties to identify the “key” moments that led them to some interpretation/conclusion, having to retrace segments of the interaction.

A single participant used the BONNIE annotation feature whilst interacting with WISE. His narrative building, therefore, was mostly guided by his own annotations. The other participants, when reminded about this feature (or were questioned about it during the interview), stated that it would have made building the narrative easier. This indicates that when faced with the added value of BONNIE, participants were open to change their interaction with SrcSys to make such annotations.

The history visualization questionnaire results distribution can be seen

in figure 7.6. The numbers in the graphs show the percentage of disagreeing answers (“1 - strongly disagree” and “2 - disagree”) on the left, neutral (“3 - neither”) at the center, and agreeing answers (“4 - agree” and “5 - strongly agree”) on the right. The sparklines on the right-hand side show the evolution of agreeing answers across the tasks (task 5 from previous study, tasks 1 and 2 from this study).

From the chart, it is possible to notice that the only statement which received the majority of disagreeing answers throughout the studies was question 7 (“I noticed the colors for the hierarchy (the gray tones) when reading it.”). Overall, all participants ignored this visual hint, but it neither had a negative impact on their performance nor on their opinions about BONNIE.

Question 9 (“It was easy to distinguish between actions that occurred in different windows.”) had interesting results, ranging from 60% to 100% to 33% agreeing rate. Out of the 5 participants, only 3 used a single window in task 2. From those 3, 2 participants did not answer question 9 and one answered it neutrally. So, the distribution was affected more by the small amount of answers than by the participants’ opinions.

The results of many questions did not change (questions 1, 6, and 10) or had little variation between studies (questions 8, 11, and 18). Most questions had increasing scores between studies (questions 4, 5, 12, 14, 15, 16, 17), especially questions 4 (“It was easy to associate what happened in the video with the representation.”, from 20% to 100%), 15 (“When collapsed, it was still easy to interpret which visualization impacts happened given a source system action.”, from 20% to 80%), and 16 (“It was easy to find the desired visualization impacts.”, from 0% to 80%).

The remaining questions showed a “V”-shaped sparkline. Question 2 (“I didn’t have to go back to the reference sheet/help to remember the meaning of some elements of the visual representation.”) ended with a 80% agreeing percentage. During the study, we did not observe the participants going back to the help material. We hypothesize that the results reflect a “neutral” posture of the participants, in the sense that they do not feel confident about knowing the system (as it got more complex).

Question 3 (“The user interaction history was easier to understand using the visual representation than only reading the descriptions.”) took a plunge during the first task, with 80% answering neutrally. We believe that seeing the video several times just before performing the task made it easier to just use the textual descriptions.

Finally, question 13 (“It was easy to find the desired source system actions.”) also had a major drop in agreement in the first task. We believe

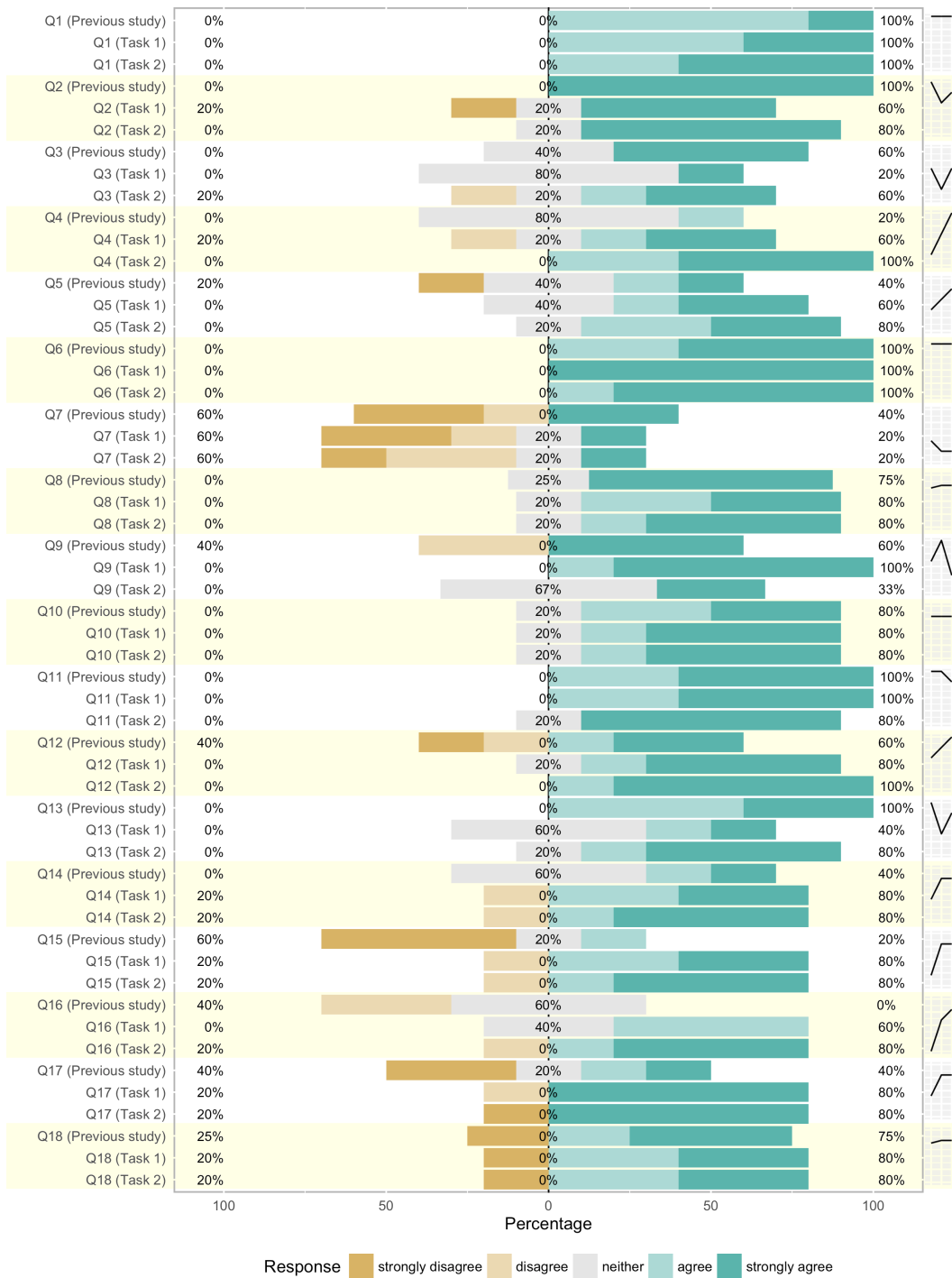


Figure 7.6: Distribution of answers to history visualization questionnaire, considering both tasks and previous study. The sparklines on the right show the evolution of “agreeing answers” percentage across tasks.

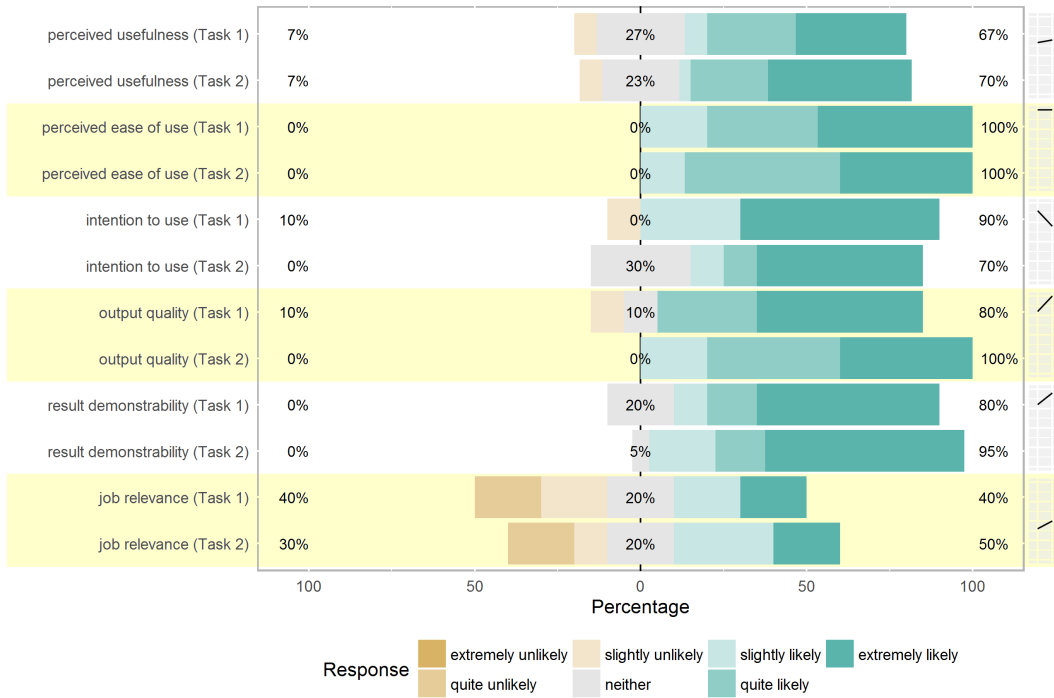


Figure 7.7: Distribution of answers to TAM questionnaire grouped by constructs and considering tasks individually. The sparklines on the right show the evolution of “agreeing answers” percentage through tasks.

that once again it was a reflection of a “neutral” stance from the participants, since the task asked specifically for the *visualization components*, not for the *source system actions*. The participants, therefore, were more focused in the *effect rows* and may have not given enough thought to the *action rows*.

Another hypothesis is that this question asked specifically about BONNIE’s model (*e.g.*, *source system actions*). The participants, not familiar with BONNIE’s model, may have hesitated in answering the questions. In future studies, we may create a new questionnaire that better frame the questions without relying on the underlying model.

Regarding the TAM questionnaires, the distribution results by constructs can be seen in figure 7.7 (grouped by task) and in figure 7.8 (considering the overall results). We adopted the 7-point Likert scale proposed by TAM. The disagreeing answers ranged from 1-3 (“extremely unlikely”, “quite unlikely”, “slightly unlikely”), 4 was the neutral one (“neither”), and 5-7 were the agreeing ones (“slightly likely”, “quite likely”, “extremely likely”). The sparklines on the right of figure 7.7 show the evolution of agreeing answers from task 1 to task 2.

The overall results were positive, with most constructs having more than 50% of agreeing answers, with the exception of “job relevance”. When filling in the questionnaire, the participants expressed difficulties in coming up with

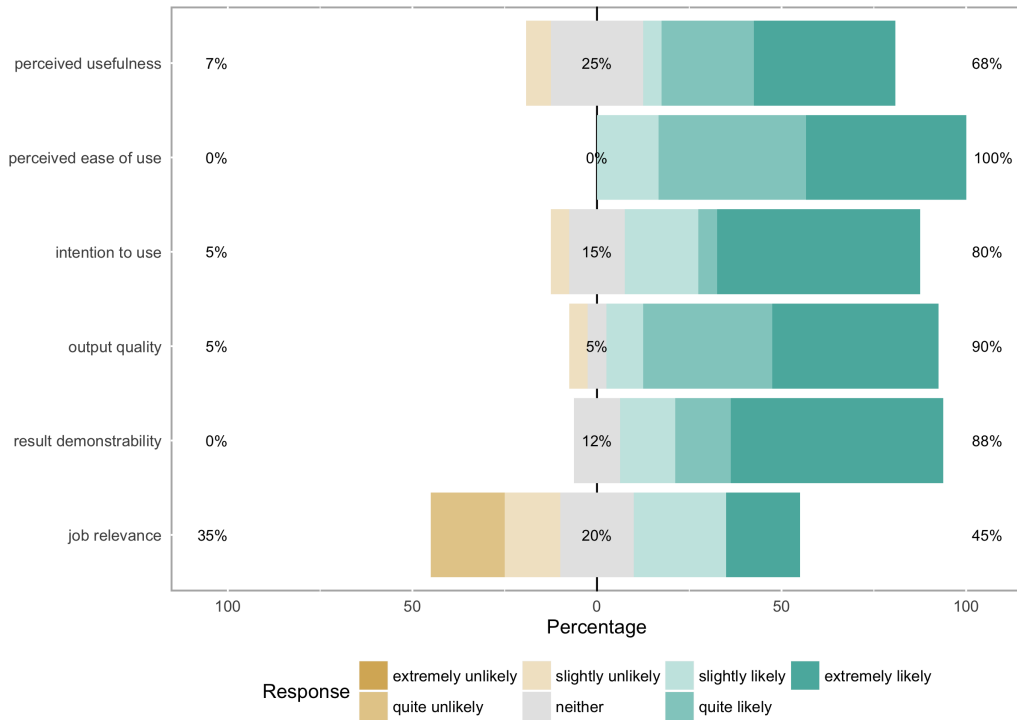


Figure 7.8: Distribution of answers to TAM questionnaire grouped by constructs, aggregating answers for both tasks.

tasks that BONNIE would help in their daily routine. During the follow-up interview, we asked their opinion about what kind of applications would better benefit from BONNIE. All participants answered something along the lines of “applications in which you perform some analysis and must create some kind of report”. This indicates that participants were able to grasp the purpose of BONNIE, although they do not see themselves using it in their day-to-day work.

A follow-up question asked participants whether the history visualization part could be useful in their daily tasks, even if they do not use the narrative builder part (since most of them told they do not have to share their reports). After some thought, most participants came with some kind of use case in which a system like BONNIE would be useful in their daily tasks, provided it could be integrated with a wider range of SrcSys (*i.e.*, not focused only on VAApps). This was an interesting feedback for the project that should be taken in consideration for future development.

Observing the results by tasks, we notice that the only construct which did not show an increase was “intention to use”. We hypothesize that this result is closely related to the “job relevance” results, since most participants could not integrate BONNIE in their daily tasks since they do not interact with VAApps as part of their jobs.

One final observation was that many participants were not aware of their own generated content. Some participants “tested” their narratives whilst building them, while others just focused on adding content. After they considered task 2 completed, the evaluator would review the participant’s narrative under the pretext of fixing the layout. During this review, many participants were surprised when what they had in mind was contrasted with their actual choices – different visualizations, different states, etc. This indicates that another interesting study would be to evaluate the generated narrative itself, both from the author’s perspective (how well the narrative fits in the author’s desired outcome) and the reader’s one (how well the narrative communicates the author’s idea).

8

Related Work

In this chapter we present some projects related to visual analytics (section 8.1), annotating visualizations (section 8.2), and data narratives (section 8.3). As Heer & Agrawala (2008) point out, there is still a large design space to be explored and many challenges remain for developing effective and scalable ways to perform collaborative data analysis.

8.1

Visual Analytics

In this section we highlight a set of tools to create a VApp (Miso in subsection 8.1.1) and a VApp that tries to figure out the best visualization for a given dataset (VizDeck in subsection 8.1.2).

8.1.1

Miso

Miso¹ is “a set of open source tools designed to make it faster and easier to create high quality interactive and data visualization content.”^{2,3} The project started in April 2012 and is still under development.

Miso consists of three tools:

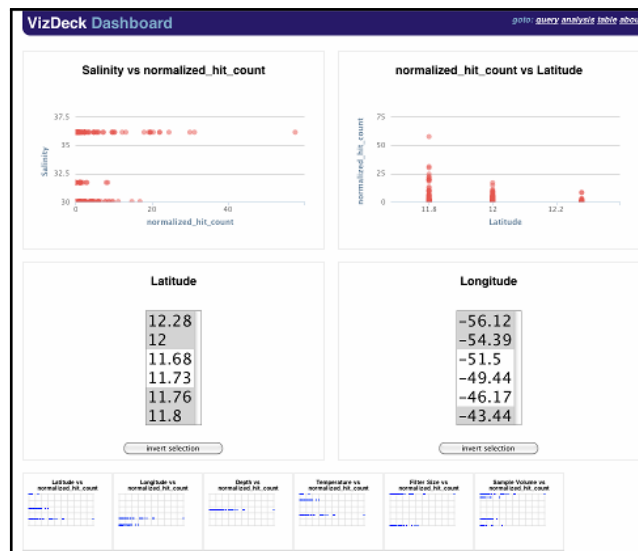
- **Dataset:** a JavaScript client-side data management and transformation library.
- **Storyboard:** a state and flow-control management library.
- **d3.chart:** a framework for creating reusable charts with d3.js.

This set of tools still relies heavily on the user developing code to create the desired data story. Our approach simplifies this process, since the user crafts the narrative based on his or her own interaction with a SrcSys and the *visualization components* already known to him or her. Users, therefore, do not need to learn how to code to be able to effectively use BONNIE.

¹<http://misoproject.com/>

²<http://www.theguardian.com/info/developer-blog/2012/apr/20/blogpost>

³<http://www.theguardian.com/news/datablog/2012/apr/21/miso-project-data-visualisation>



Adapted from <http://escience.washington.edu/vizdeck>

Figure 8.1: VizDeck dashboard interface.

8.1.2 VizDeck

VizDeck⁴ is “a web-based tool for exploratory visual analytics of unorganized relational data” (Key et al., 2012). Together with SQLShare,⁵ they are a solution that allows managing data from various heterogeneous sources, querying, and visually analyzing them.

It works as a chart recommender based on the statistical properties of the data. With a card metaphor, it allows the user to organize the proposed visualizations into interactive dashboards that can be shared later.

Its main limitation is the lack of comment and annotation features, so the knowledge discovered in a dashboard cannot be explicitly shared. Also, since it does not target a specific data domain, its visualization recommendations rely only on some statistical features of the data (number of distinct values, entropy, coefficient of variation, kurtosis, and periodicity), which can be misleading.⁶

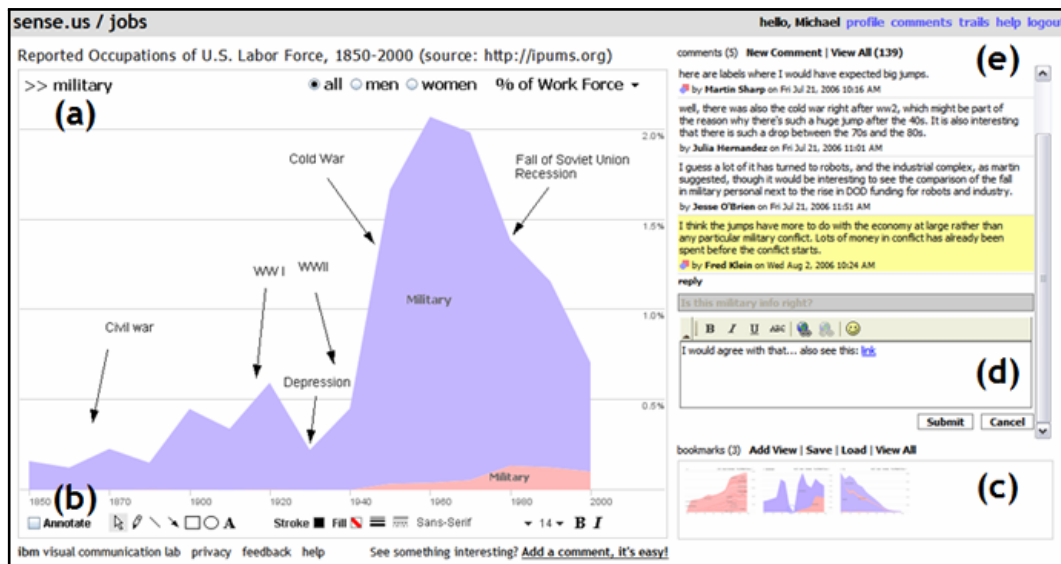
Regarding visual analytics, VizDeck’s approach follows a more “linear”/turn-based sequence: the “computer” suggests charts and the “human” selects one of them. There is no seamless integration, with the “computer” empowering the “human” to aid his or her decision-making skills.

Also, creating a dashboard considering only the currently available data may “hide” some information. For example, if the currently available data do not present some relevant statistical property, the corresponding “best”

⁴<http://escience.washington.edu/vizdeck>

⁵<http://escience.washington.edu/sqlshare>

⁶Anscombe’s quartet (https://en.wikipedia.org/wiki/Anscombe%27s_quartet) is a famous example where four datasets have nearly the same statistical properties but plot very differently in a scatterplot.



Adapted from <http://vis.stanford.edu/papers/senseus>

Figure 8.2: The sense.us collaborative visualization system.

visualization (from the user's point of view) would get a lower ranking score, "hiding" it from the user. The dashboard would then be created without this visualization, one that could be important when new data arrive.

8.2 Annotating Visualizations

In this section, we discuss some projects that take a first step towards a narrative, allowing to annotate a single visualization. Sense.us (subsection 8.2.1) and ManyEyes (subsection 8.2.2) allow the user to create annotations manually, exploring the social and community aspects of multiple users annotating the same visualization. Contextifier (subsection 8.2.3) creates automatic annotations, linking the visualization with a news article database.

8.2.1 sense.us

Sense.us (Heer et al., 2007) is a research prototype that focuses on visualization annotations and building tours through multiple visualization states (Heer & Agrawala, 2008). This focus explains its main limitation: the only *visualization component* is a stacked area chart (figure 8.2 a) showing a set of demographic data, with some filtering features. On the right-hand side panel it is possible to see the comments already associated to the current visualization (figure 8.2 e) and to add new comments (figure 8.2 d), which can be improved by the annotation tools (figure 8.2 b) and by linking them to other saved visualizations (figure 8.2 c).

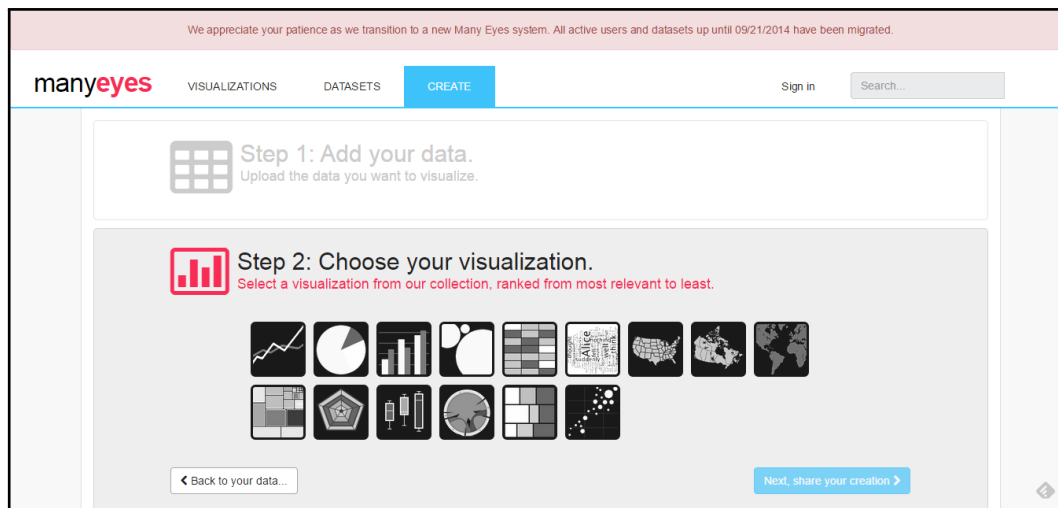


Figure 8.3: Many Eyes interface.

8.2.2

Many Eyes

Many Eyes⁷ is a public website that allows the creation of dataset visualizations. The user first chooses (or creates) a dataset and then a visualization from a ranked list of available choices for the type of data in the dataset. Then, the user can configure some visual parameters and finally share both the dataset and the visualization with the community.

The main characteristic of Many Eyes is the social aspect of the created visualizations, with public datasets and visualizations, which can receive comments from the community. This focus on community-generated content happens to also be its main weakness: for example, many of the available content is labeled as “test”, indicating that it was created by a user “playing” with the tool. Also, it seems that there is no way to relate two different visualizations from distinct datasets besides writing some comments in both visualizations.

8.2.3

Contextifier

Contextifier (Hullman et al., 2013) is a “system that automatically produces custom, annotated visualizations of stock behavior given a news article about a company.” It combines a news article corpus, a query generator, an annotation selection engine, and a graph generator to produce an annotated stock market visualization (figure 8.4).

It has only a single visualization. It focuses more on the automatic detection of interesting data points and cross-referencing them with news

⁷www.manyeyes.com

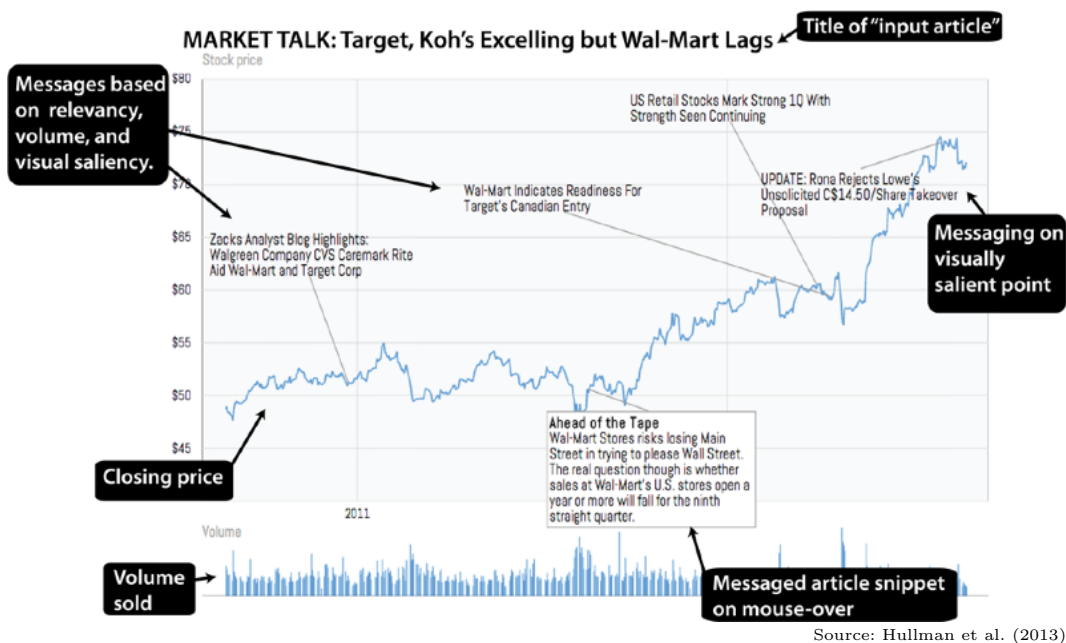


Figure 8.4: A visualization produced by Contextifier with some callouts to interface features.

articles than to empower the user to express his or her own narrative.

8.3 Data Narratives

The projects in this section allow the creation of narratives containing visualizations. SketchStory (subsection 8.3.1) allows the user to draw visualizations while presenting his or her narrative. Ellipsis (subsection 8.3.2) and Tableau (subsection 8.3.3) offer an environment to create visualizations, define their parameters, coordinate them, and create an interactive narrative. Finally, VisTrails (subsection 8.3.4) is an environment to create visualizations from a workflow and explore the provenance data from the workflow definition process. The narrative can be inferred by the user's own interaction history visualization, but is not explicit at the UI. In all of these projects, the user is the creator of both the visualization and the narrative, as opposed to our solution, in which the user is the consumer of a visualization designed by the SrcSys development team and the creator of a narrative using BONNIE.

8.3.1 SketchStory

SketchStory (Lee et al., 2013) is "a data-enabled digital whiteboard that facilitates the creation of personalized and expressive data charts quickly and easily." It allows presenters to create charts during the presentation using pen and touch interactions, as illustrated in figure 8.5.



Source: Lee et al. (2013)

Figure 8.5: Telling a story using SketchStory: (left) A presenter sketches out an example icon and chart axis, (middle) Upon recognition of the chart axis, SketchStory completes the chart with underlying data by synthesizing from example sketches, and (right) The presenter interacts with the charts.

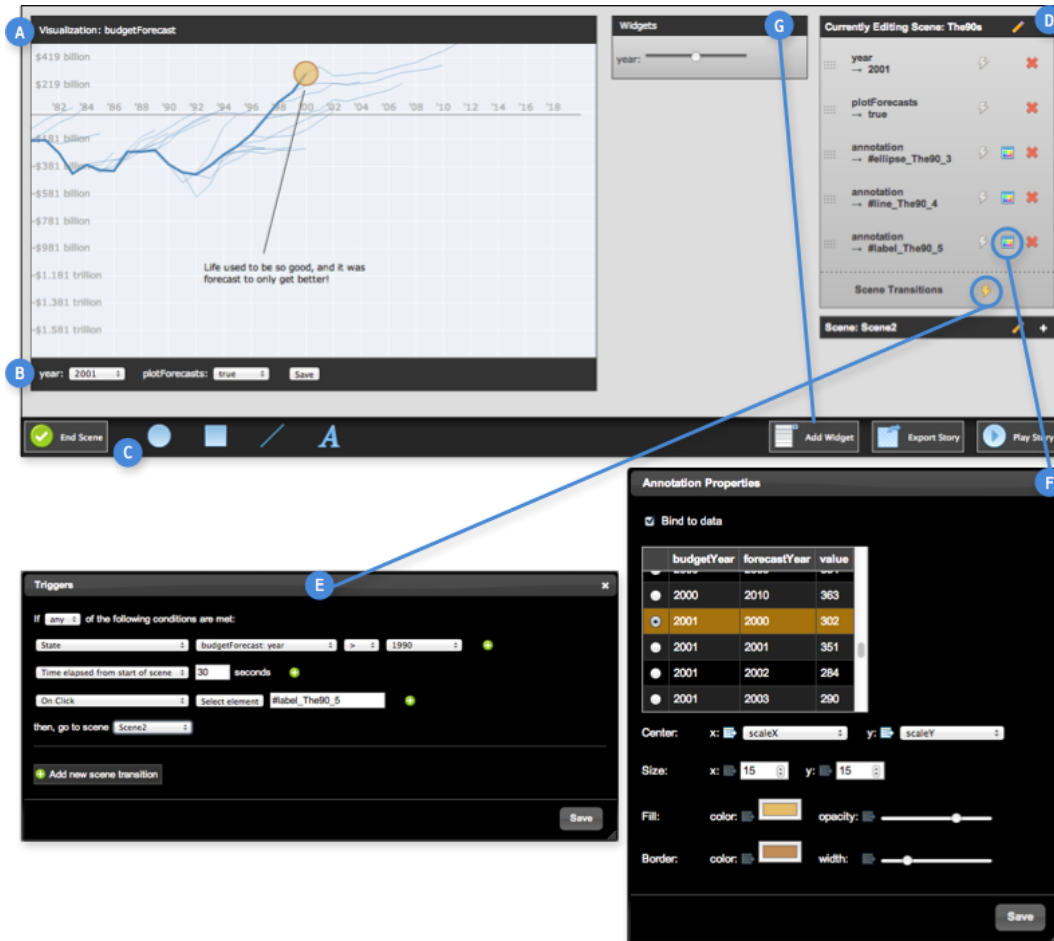
SketchStory focuses on real-time synchronous presentation, with the presenter telling his narrative while presenting to the consumers. Our solution had another goal: to output a narrative that could be consumed asynchronously, possibly with some reader interaction.

8.3.2 Ellipsis

Ellipsis (Satyanarayan & Heer, 2014) is a system for creating narrative visualizations. It is comprised by a domain-specific language and a web-based interface (shown in figure 8.6). In the UI, the user can manipulate scenes, annotations, and user interactions directly. The underlying domain-specific language is update accordingly to changes made in the UI.

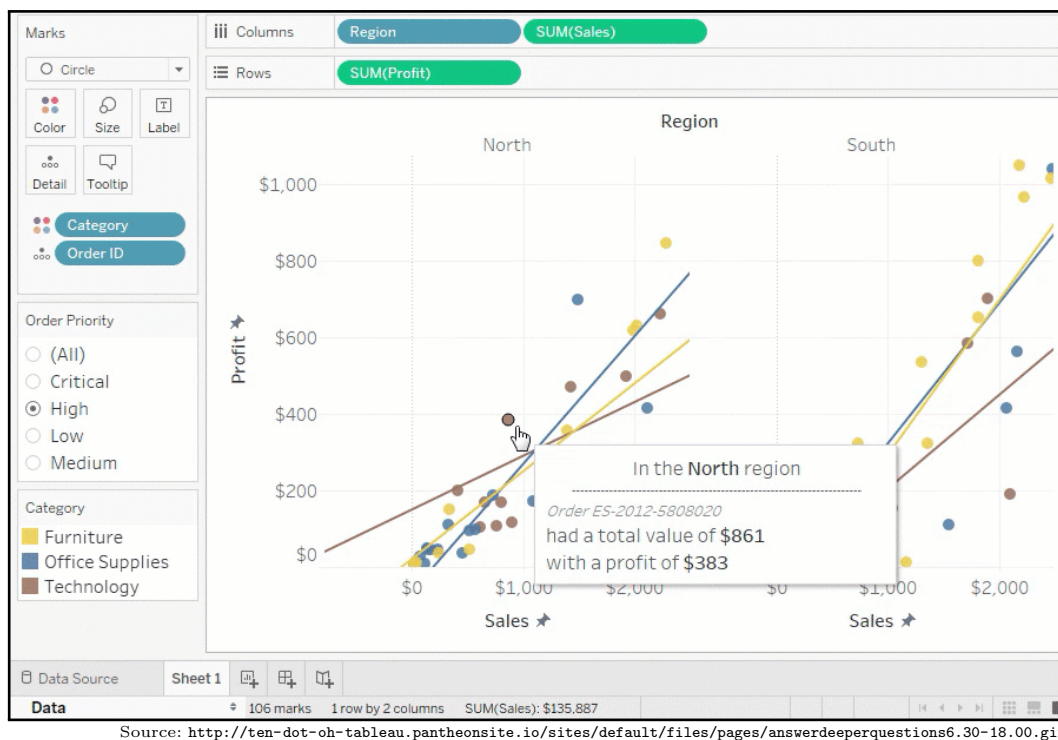
Ellipsis define a narrative visualization as “a set of visualization components, control widgets and annotations that are coordinated by a narrative state machine.” Our solution presents similar concepts, with the exception of the “control widgets” and, thus, the necessity of a “narrative state machine”. Since we aimed to have a more author-driven narrative with a linear flow, these concepts were not needed. We, however, plan to further extend our narrative building features, so we could create narratives along the author/reader-driven spectrum.

Moreover, the authors describe their core abstractions as being “state-based scenes, visualization parameters, dynamic graphical & textual annotations, and interaction triggers.” Once again, we have some similarities, since our *panels* is similar to their scenes, and our *visualization* and *textual elements* to their graphical and textual annotations. We currently do not support visualization parameters and interaction triggers, since our focus is not the coordination between *visualization elements*.



Source: Satyanarayan & Heer (2014)

Figure 8.6: The Ellipsis interface. (a) Ellipsis creates a stage element for each visualization. (b) The GUI inspects visualization parameters and creates controls for them. (c) Creating a new scene prompts the storyteller for a scene name; scenes can be built by changing visualization parameters or drawing annotations. (d) The sidebar lists reorderable scenes and members. (e) Triggers and scene transitions are defined using an “if this, then that” syntax. (f) Annotation properties and data binding can be modified, triggering real-time updates. (g) Standard form widgets can be instantiated and bound to visualization parameters.



Source: <http://ten-dot-oh-tableau.pantheonsite.io/sites/default/files/pages/answerdeeperquestions6.30-18.00.gif>

Figure 8.7: Example of Tableau interface.

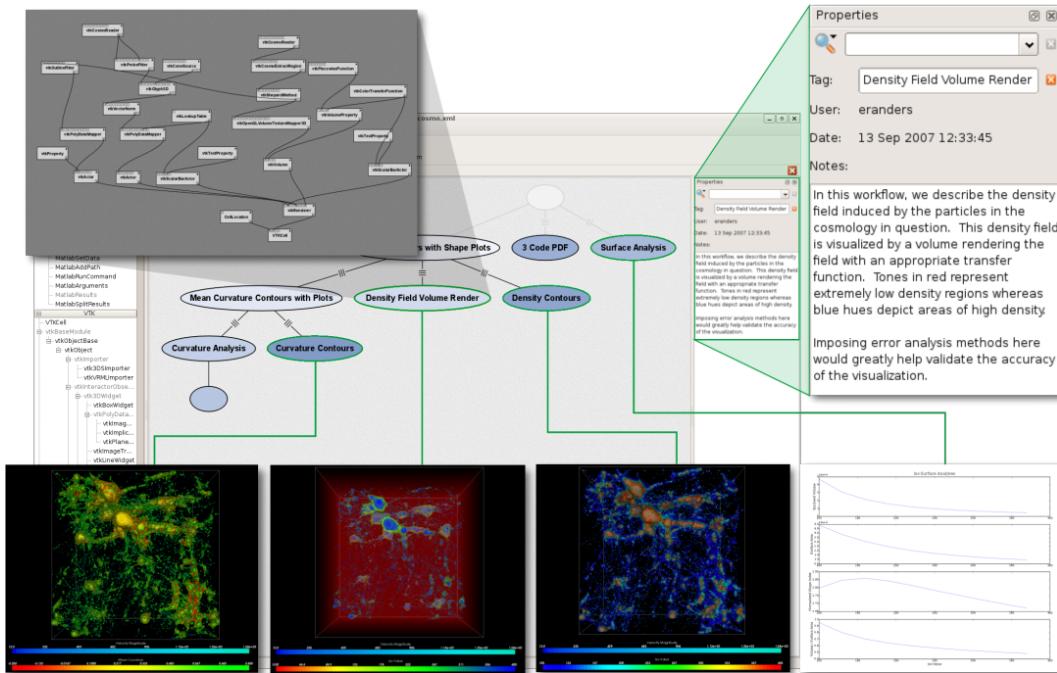
The main difference to our solution is the moment and context in which the user interacts with the visualization. With Ellipsis, the user interacts with the visualization “inside” Ellipsis, defining parameters and coordinating multiple visualizations. As aforementioned, the user is the author of not only the narrative, but also of the visualizations. Our solution considers that the interaction with the visualizations happens in SrcSys. The configuration and coordination of visualizations, therefore, is a SrcSys’s developer responsibility. The user is the consumer of the visualization, interpreting it, and gaining insights. This perspective led us to explore the trace left by the interaction with SrcSys to create a narrative.

8.3.3 Tableau

Tableau⁸ is a company “on a mission to help people see and understand data.”⁹ Its roots are in research conducted in Stanford University, such as the Visual Query Language (VizQL) – a database visualization language, combining a structured query language for databases with a descriptive language for rendering graphics – and Polaris – an interface for exploring large multi-dimensional databases.

⁸<http://www.tableausoftware.com/>

⁹<http://www.tableausoftware.com/about>



Source: http://www.vistrails.org/images/Cosmology_example.png

Figure 8.8: VisTrails interface with highlights.

PUC-Rio - Certificação Digital Nº 1212408/CA

It now commercializes a suite of products (Tableau Desktop, Tableau Server, Tableau Online, Tableau Reader, and Tableau Public) which aims to make databases and spreadsheets understandable to ordinary people. The products can query different sorts of databases – relational, cubes, cloud, and spreadsheets – and then generate a number of visualizations that can be arranged in dashboards and shared.

The focus is on the visual analytics aspect, with powerful tools to explore the databases and create the dashboards. It also allows to create interactive presentations, with some degree of reader freedom.

Similar to Ellipsis, the user is both the creator of visualization and the author of narratives. In our work, the user is a consumer of the visualizations from the SrcSys and the author of a narrative based on them and his or her interaction with them.

8.3.4 VisTrails

VisTrails (Santos et al., 2009; Silva et al., 2011) is “an open-source scientific workflow and provenance management system that supports data exploration and visualization.”¹⁰ It integrates with different libraries (*e.g.*, VTK, Matplotlib, NumPy, SciPy) to create workflows that generate visualizations.

¹⁰<http://www.vistrails.org/>

Figure 8.8 shows an example of some visualizations created with VisTrails. The actual vistrail – the history tree – is at the center of the figure. Each node in this tree corresponds to a visualization (shown at the bottom), which is associated with a workflow (shown on the top left). The edges indicate changes between nodes, a transformation from the parent node to the child node.

VisTrails focuses on the provenance of the workflow to create a single visualization. Its history visualization, therefore, is able to compare the changes between different workflows. Given this objective, the tree visualization is good to express the undo/redo and the transformation hierarchy.

BONNIE aims to integrate with different *source systems* that may have different *views* and *visualization components*, each one with different interactivity. We do not have the underlying information of how the visualizations are generated to be able to compare them. Our history visualization, therefore, focuses on the temporal sequence of *source system actions* being a linear visualization instead of a tree one.

8.4

Concluding Remarks

Despite an extensive literature review, we could not find a solution similar to BONNIE. Many systems have an integrated history manager (*e.g.*, sense.us, Tableau, VisTrails), but few of them show the history from other systems. Moreover, we could not find one that integrates interactive history visualization with interactive creation of a narrative (*e.g.*, Miso provides a development tool to manually code a storyboard).

9 Conclusion

Would you tell me, please, which way I ought to go from here?

That depends a good deal on where you want to get to.

I don't much care where.

Then it doesn't matter which way you go.

Lewis Carroll, *Alice's Adventures in Wonderland* (1865)

As the “three V’s” (data volume, velocity, and variety) increase, VAApps become even more important. Combining the processing power of the computer and the human cognitive abilities allows the exploration and, more importantly, the interpretation of such data. Since the main purpose of visualization is insight (Card et al., 1999, p 6), the communication of this acquired knowledge is of utmost importance.

In their work regarding graph comics, Bach et al. (2016) state that a lightweight editor – allowing the creation and layout of panels; adding, positioning, and customizing elements; and adding textual annotations – would greatly improve the authoring process. In this thesis, we presented BONNIE, a framework that enables users to create an interactive narrative from previous logged interaction events from another VAApp – the SrcSys.

The next section (section 9.1) summarizes our main contributions. Section 9.2 highlights next steps regarding BONNIE research.

9.1 Contributions

As stated in the Introduction, this thesis addressed the research question “*How can we provide support for users to communicate their findings based on their interaction history with a VAApp?*” and the “sub-questions”:

1. *What should we capture and how to save interaction events from a VAApp?*
2. *How should we change the VAApp to make it compatible with our solution?*

3. *How can we display the user's interaction history from another VAApp?*
4. *How to enable the user to create a narrative from his or her interaction history?*

Considering the first sub-question, we defined a log model that enables the recording of multiple interaction events, from opening a new window, navigating to a page, a domain-specific action, and a domain-independent *visualization effect*. Moreover, we proposed a way to store the information following the PROV-XML schema.

To address the second sub-question, we created auxiliary code to help the SrcSys developers to instrument their applications. Besides logging, this auxiliary code comprehends abstract classes to some SrcSys components (namely, *visualization components* and *data services*), so they can be later used by BONNIE. We implemented this solution using WISE as a study case, gathering instructions on how to better instrument and organize the SrcSys application.

Finally, we developed BONNIE UI, creating a “lightweight” narrative editor and tackling the last two sub-questions. We present the history using an interactive timeline visualization, allowing the user to explore his or her own interaction events history. With these events, the user can create a narrative, adding visualization and textual elements to panels, by dragging-and-dropping elements from the history visualization.

Besides the actual implementation, we performed three separate studies. The first was an analytical study of the history visualization, using the Physics of Notation (Moody, 2009) and the Cognitive Dimensions of Notation framework (Green & Petre, 1996). We furthered the history visualization analysis with a user study, having participants go through several tasks to try to uncover their mappings from the SrcSys domain to the BONNIE domain. Finally, we conducted a second user study considering the whole system and a narrative creation task. The studies' results show BONNIE to be a promising solution to address our main research question.

9.2

Future Work

Many basic features are still missing in BONNIE's UI (a small list can be found in appendix G); for example, filtering the visualization by *view*, *source system action*, or *visualization component*.

Although the studies yielded, in general, a positive outcome, a major limitation of our results is that, given time and resource constraints, only WISE

was instrumented during the development of the thesis. There were several iterations of the framework – taking into consideration input from different development teams – which made unfeasible to instrument other SrcSys.

WISE was the test bed needed to evolve our solution. Given that we reached a stable solution, we plan to invite other developers to instrument their own systems. This will start a new development cycle, with new feedback and, possibly, new requirements that we have not uncovered in our work.

Further evaluations of the UI, *e.g.*, a heuristic evaluation (Nielsen, 1993), could reveal even more missing features. We can also use the communicability evaluation method (De Souza & Leitão, 2009) to further investigate the communication breakdowns in the interaction between the final user, the SrcSys designer, and the BONNIE designer (as shown in figure 6.10). Such a study could lead to a set of guidelines to improve BONNIE’s communicability.

We can also explore other output means for the logged data in addition to our history visualization. For example, we can consider playing an animation from the saved interaction events so the user could have a “playback” of his or her interaction with SrcSys.

We plan to change the implementation of the framework to cause less impact on the SrcSys. Currently, whilst interacting with SrcSys in the browser, the loaded page makes several post requests to the *history log data service*. An initial idea is to develop a desktop application so the web page communicates with this desktop application. This would bring some benefits, for example:

- No additional HTTP requests — With the current implementation, the interaction events logs are posted to the *history log data service* whilst the user interacts with the page. This may impact in the SrcSys performance, given that these post requests are queued together with the SrcSys regular calls. Calling a desktop application would free up the queue, reducing the impact on SrcSys.
- Save the interaction events log locally — If the user navigates to another web page or the browser is closed, the logs can be lost. We could still use the browser’s cache, but this would depend on the user accessing the SrcSys again. Using the desktop application, the interaction events logs can be saved to the user’s hard drive and sent to the *history log data service* at an appropriate time.
- Better user *session* — We envision BONNIE to gather data from multiple SrcSys simultaneously. One issue of this idea is how to share user information amongst SrcSys without impacting their development (*e.g.*, relying on a shared user database). Our current log model implementation considers that the `user` has a `localId` associated with the SrcSys

and a `globalId` shared amongst SrcSys and managed by BONNIE. This approach may not be feasible, since it would rely on keeping multiple user databases in sync and adding the requirement of having a login system in the SrcSys. With the desktop application, the user would login in the application using BONNIE credentials (solely managed by BONNIE). The SrcSys would post to the desktop application and, therefore, use the BONNIE credentials. This would allow having independent users between different SrcSys and BONNIE, whilst also not relying on a SrcSys login.

We also plan to simplify the PROV-XML documents, using fewer elements and changing the restrictions. We are considering using a variable collection of *visualization components*, so a given `View` may not need to reference a `ViewDef`. This could also impact the *definition hierarchy*, maybe with the `SrcSysDef` having a collection of `VisCompDef` instead of `ViewDef`.

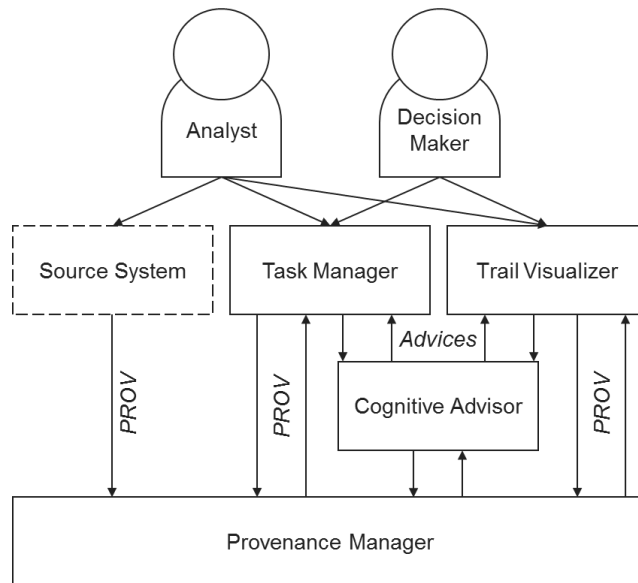
Another idea is to use the collected log data and the created narratives to create an *importance model* for the logged interaction events. With this *importance model*, we could highlight or subdue certain steps in the *history visualization*, making it easier for users to find the events. Moreover, we could further develop this *importance model* so we can propose a starting narrative when the user opens BONNIE.

We will integrate BONNIE with the Cognitive Analytic Trail System (CATS) (Thiago et al., 2016). CATS is a system for capturing provenance data from a SrcSys, integrating it with a user “task manager”. A “cognitive advisor” uses the provenance and task data to provide advices to the user while interacting with SrcSys. The CATS’s architecture can be seen in figure 9.1.

BONNIE will be its “trail visualizer” component, displaying data from the “provenance manager” (similar to the *user history log* from BONNIE’s architecture). Moreover, additional information will be obtained from the other components (“task manager” and “cognitive advisor”), allowing us to enhance our UI.

CATS can also bring about new use cases for BONNIE. For example, we can think about one user viewing the interaction history of a second user (*e.g.*, the decision maker viewing the analyst’s interaction history). Another use case would be to detect similar interaction patterns, allowing to compare different interaction sequences (*e.g.*, compare the interaction sequence from two analysts with the same task) or even helping the user interacting with the SrcSys by suggesting the next step in the detected interaction pattern.

Regarding the narrative builder, we can export to other formats, such as a static slide show (*e.g.*, a Power Point presentation), a static document (*e.g.*,



Source: Thiago et al. (2016)

Figure 9.1: CATS architecture.

a PDF document), or a multimedia document (*e.g.*, using NCL¹). We can also further explore the narrative aspect, providing more elements to add to a given panel (*e.g.*, arrows and call-outs) and different transitions (McCloud, 1993). We can also explore how to build non-linear narratives (*e.g.*, also using NCL), creating different paths to the reader in a hypermedia document.

Besides the “slide show-like” builder environment, we can study a visualization for the narrative structure being created. For example, we could use the Rhetorical Structure Theory² to organize the narrative concepts so we can suggest a narrative or even highlight discrepancies between the structure and the narrative being created.

Last but not least, with the proposed framework we envision the possibility of creating a meta-environment for generating VAApps. Given that we have *visualization component* and *data service* definitions, we could create an environment to connect them to generate a new VAApp. We would have a *visualization component* and *data service* library, from which the user could choose a *data service* call to gather data to encode in a *visualization component*. We could also create the “event” concept, so an action that happens in a *visualization component* could change another one. The *visualization component* and *data service* base class interface would become more complex, eliciting more details (such as the “event”, how to “encode” data, which data are expected, etc.) to guarantee compatibility between them.

¹<http://www.ncl.org.br/en>²<http://www.sfu.ca/rst/>

10

References

- Aigner, W., Bertone, A., Miksch, S., Tominski, C. & Schumann, H. (2007a), Towards a conceptual framework for visual analytics of time and time-oriented data, *in* ‘Simulation Conference, 2007 Winter’, pp. 721–729. DOI 10.1109/WSC.2007.4419666.
- Aigner, W., Miksch, S., Müller, W., Schumann, H. & Tominski, C. (2007b), ‘Visualizing time-oriented data – a systematic view’, *Computers & Graphics* **31**(3), 401–409. DOI 10.1016/j.cag.2007.01.030. ISSN 0097-8493. URL <http://www.sciencedirect.com/science/article/pii/S0097849307000611>.
- Andrienko, G., Andrienko, N., Keim, D., MacEachren, A. M. & Wrobel, S. (2011), ‘Editorial: Challenging problems of geospatial visual analytics’, *J. Vis. Lang. Comput.* **22**(4), 251–256. DOI 10.1016/j.jvlc.2011.04.001. ISSN 1045-926X.
- Atterer, R. & Schmidt, A. (2007), Tracking the interaction of users with ajax applications for usability testing, *in* ‘Proceedings of the SIGCHI Conference on Human Factors in Computing Systems’, CHI ’07, ACM, New York, NY, USA, pp. 1347–1350. DOI 10.1145/1240624.1240828.
- Atterer, R., Wnuk, M. & Schmidt, A. (2006), Knowing the user’s every move: User activity tracking for website usability evaluation and implicit interaction, *in* ‘Proceedings of the 15th International Conference on World Wide Web’, WWW ’06, ACM, New York, NY, USA, pp. 203–212. DOI 10.1145/1135777.1135811.
- Bach, B., Kerracher, N., Hall, K. W., Carpendale, S., Kennedy, J. & Henry Riche, N. (2016), Telling stories about dynamic networks with graph comics, *in* ‘Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems’, CHI ’16, ACM, New York, NY, USA, pp. 3670–3682. DOI 10.1145/2858036.2858387.
- Benbunan-Fich, R., Hiltz, S. R. & Turoff, M. (2003), ‘A comparative content analysis of face-to-face vs. asynchronous group decision making’, *Decision*

Support Systems **34**(4), 457–469. DOI 10.1016/S0167-9236(02)00072-6. ISSN 0167-9236. URL <http://www.sciencedirect.com/science/article/pii/S0167923602000726>.

Bertin, J. (1983), ‘Semiology of graphics: Diagrams, networks, maps’.

Blackwell, A. & Green, T. (2003), Notational systems – the cognitive dimensions of notations framework, *in* J. Carroll, ed., ‘HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science’, Interactive Technologies, Elsevier Science, chapter 5, pp. 103–134. URL <https://books.google.com.br/books?id=gGyE0jkdpyC>.

Brehmer, M. & Munzner, T. (2013), ‘A multi-level typology of abstract visualization tasks’, *Visualization and Computer Graphics, IEEE Transactions on* **19**(12), 2376–2385. DOI 10.1109/TVCG.2013.124. ISSN 1077-2626.

Card, S. K., Mackinlay, J. D. & Shneiderman, B. (1999), *Readings in Information Visualization: Using Vision to Think*, Morgan Kaufmann.

Chung, W., Chen, H., Chaboya, L. G., O’Toole, C. D. & Atabakhsh, H. (2005), ‘Evaluating event visualization: a usability study of COPLINK spatio-temporal visualizer’, *International Journal of Human-Computer Studies* **62**(1), 127–157. DOI 10.1016/j.ijhcs.2004.08.005. ISSN 1071-5819. URL <http://www.sciencedirect.com/science/article/pii/S1071581904001004>.

Cockburn, A., Karlson, A. & Bederson, B. B. (2008), A review of overview+detail, zooming, and focus+context interfaces, *in* ‘ACM Surveys’, Association for Computing Machinery, Inc. URL <https://www.microsoft.com/en-us/research/publication/a-review-of-overviewdetail-zooming-and-focuscontext-interfaces/>.

Cook, K. A. & Thomas, J. J. (2005), Illuminating the path: The research and development agenda for visual analytics, Technical report, Pacific Northwest National Laboratory (PNNL), Richland, WA (US).

Cook, K., Earnshaw, R. & Stasko, J. (2007), ‘Guest editors’ introduction: Discovering the unexpected’, *Computer Graphics and Applications, IEEE* **27**(5), 15–19. DOI 10.1109/MCG.2007.126. ISSN 0272-1716.

Davis, F. D. (1989), ‘Perceived usefulness, perceived ease of use, and user acceptance of information technology’, *MIS Quarterly* **13**(3), 319–340. ISSN 02767783. URL <http://www.jstor.org/stable/249008>.

- de Sousa, T. A. F. & Barbosa, S. D. J. (2012), Semantic characterization of visualization mechanisms, *Monografias em Ciência da Computação* 16, PUC-Rio.
- de Sousa, T. A. F. & Barbosa, S. D. J. (2013), Sistema de recomendação para apoiar a construção de gráficos com dados estatísticos, *in* 'Proceedings of the 12th Brazilian Symposium on Human Factors in Computing Systems', IHC '13, Brazilian Computer Society, Porto Alegre, Brazil, Brazil, pp. 168–177. URL <http://dl.acm.org/citation.cfm?id=2577101.2577136>.
- De Souza, C. & Leitão, C. (2009), *Semiotic Engineering Methods for Scientific Research in HCI*, Synthesis lectures on human-centered informatics, Morgan & Claypool. URL <https://books.google.com.br/books?id=6JCwq-CGRwAC>.
- de Souza, C. S. (2005), *The Semiotic Engineering of Human-Computer Interaction (Acting with Technology)*, The MIT Press.
- Doleisch, H. (2007), SimVis: Interactive visual analysis of large and time-dependent 3D simulation data, *in* 'Simulation Conference, 2007 Winter', pp. 712–720. DOI 10.1109/WSC.2007.4419665.
- Elias, M., Aufaure, M.-A. & Bezerianos, A. (2013), Storytelling in visual analytics tools for business intelligence, *in* P. Kotzé, G. Marsden, G. Lindgaard, J. Wesson & M. Winckler, eds, 'Human-Computer Interaction – INTERACT 2013', Vol. 8119 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 280–297.
- Ferreira, J. J., de Souza, C. S., de Castro Salgado, L. C., Slaviero, C., Leitão, C. F. & d. F. Moreira, F. (2012), Combining cognitive, semiotic and discourse analysis to explore the power of notations in visual programming, *in* 'Visual Languages and Human-Centric Computing (VL/HCC), 2012 IEEE Symposium on', pp. 101–108. DOI 10.1109/VLHCC.2012.6344492. ISSN 1943-6092.
- Ferreira, J. S. J., Segura, V. & Cerqueira, R. (2015), Cognitive dimensions of notation tailored to environments for visualization and insights, *in* 'Proceedings of the XIV Brazilian Symposium on Human Factors in Computer Systems', IHC 2015.
- Few, S. (2013), 'Data visualization for human perception', *The Encyclopedia of Human-Computer Interaction*, 2nd Ed.

- Genon, N., Heymans, P. & Amyot, D. (2011), Analysing the cognitive effectiveness of the BPmn2.0 Visual notation, *in* B. Malloy, S. Staab & M. Brand, eds, 'Software Language Engineering: Third International Conference, SLE 2010, Eindhoven, The Netherlands, October 12-13, 2010, Revised Selected Papers', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 377–396.
- Green, M. J. & Myers, K. R. (2010), 'Graphic medicine: Use of comics in medical education and patient care', *BMJ*. DOI 10.1136/bmj.c863. ISSN 0959-8138. URL <http://www.bmj.com/content/340/bmj.c863>.
- Green, T. R. G. & Petre, M. (1996), 'Usability analysis of visual programming environments: A 'cognitive dimensions' framework', *Journal of Visual Languages & Computing* **7**(2), 131–174. DOI 10.1006/jvlc.1996.0009. ISSN 1045-926X. URL <http://www.sciencedirect.com/science/article/pii/S1045926X96900099>.
- Green, T., Ribarsky, W. & Fisher, B. (2008), Visual analytics for complex concepts using a human cognition model, *in* 'Visual Analytics Science and Technology, 2008. VAST '08. IEEE Symposium on', pp. 91–98. DOI 10.1109/VAST.2008.4677361.
- Heer, J. & Agrawala, M. (2008), 'Design considerations for collaborative visual analytics', *Information Visualization* **7**(1), 49–62. DOI 10.1057/palgrave.ivs.9500167. URL <http://ivi.sagepub.com/content/7/1/49.abstract>.
- Heer, J. & Shneiderman, B. (2012), 'Interactive dynamics for visual analysis', *Queue* **10**(2), 30:30–30:55. DOI 10.1145/2133416.2146416. ISSN 1542-7730.
- Heer, J., Bostock, M. & Ogievetsky, V. (2010), 'A tour through the visualization zoo', *Commun. ACM* **53**(6), 59–67. DOI 10.1145/1743546.1743567. ISSN 0001-0782.
- Heer, J., Viégas, F. B. & Wattenberg, M. (2007), Voyagers and voyeurs: Supporting asynchronous collaborative information visualization, *in* 'Proceedings of the SIGCHI conference on Human factors in computing systems', ACM, pp. 1029–1038.
- Hullman, J., Diakopoulos, N. & Adar, E. (2013), Contextifier: Automatic generation of annotated stock visualizations, *in* 'Proceedings of the SIGCHI Conference on Human Factors in Computing Systems', CHI '13, ACM, New

York, NY, USA, pp. 2707–2716. DOI 10.1145/2470654.2481374. URL <http://doi.acm.org/10.1145/2470654.2481374>.

IBM (2012), ‘2012 IBM annual report’. URL http://www.ibm.com/annualreport/2012/bin/assets/2012_ibm_annual.pdf.

IBM (2013), ‘2013 IBM annual report’. URL http://www.ibm.com/annualreport/2013/bin/assets/2013_ibm_annual.pdf.

Jun, E., Landry, S. & Salvendy, G. (2011), ‘A visual information processing model to characterize interactive visualization environments’, *Intl. Journal of Human–Computer Interaction* **27**(4), 348–363.

Keim, D. A., Kohlhammer, J., Ellis, G. & Mansmann, F. (2010), *Mastering the Information Age - Solving Problems with Visual Analytics*, Florian Mansmann. URL <http://books.google.com.br/books?id=vdv5wZM8ioIC>.

Keim, D. A., Mansmann, F., Oelke, D. & Ziegler, H. (2008), Visual analytics: Combining automated discovery with interactive visualizations, in ‘Proceedings of the 11th International Conference on Discovery Science’, DS ’08, Springer-Verlag, Berlin, Heidelberg, pp. 2–14. DOI 10.1007/978-3-540-88411-8_2.

Keim, D., Mansmann, F., Schneidewind, J., Ziegler, H. et al. (2006), Challenges in visual data analysis, in ‘Information Visualization, 2006. IV 2006. Tenth International Conference on’, IEEE, pp. 9–16.

Key, A., Howe, B., Perry, D. & Aragon, C. (2012), Vizdeck: Self-organizing dashboards for visual analytics, in ‘Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data’, SIGMOD ’12, ACM, New York, NY, USA, pp. 681–684. DOI 10.1145/2213836.2213931.

Knafllic, C. N. (2015), *Storytelling with Data: A Data Visualization Guide for Business Professionals*, Wiley. URL <https://books.google.com.br/books?id=retRCgAAQBAJ>.

Kohlhammer, J., Keim, D., Pohl, M., Santucci, G. & Andrienko, G. (2011), ‘Solving problems with visual analytics’, *Procedia Computer Science* **7**(0), 117–120. DOI 10.1016/j.procs.2011.12.035. ISSN 1877-0509. URL <http://www.sciencedirect.com/science/article/pii/S1877050911007009>, jce:title;Proceedings of the 2nd European Future Technologies Conference and Exhibition 2011 (FET 11);/ce:title;.

- Lazar, J., Feng, J. H. & Hochheiser, H. (2010), *Research Methods in Human-Computer Interaction*, John Wiley & Sons Ltd.
- Lee, B., Kazi, R. H. & Smith, G. (2013), ‘Sketchstory: Telling more engaging stories with data through freeform sketching’, *IEEE Transactions on Visualization and Computer Graphics* **19**(12), 2416–2425. DOI 10.1109/TVCG.2013.191. ISSN 1077-2626.
- Ma, K.-L., Liao, I., Frazier, J., Hauser, H. & Kostis, H.-N. (2012), ‘Scientific storytelling using visualization’, *Computer Graphics and Applications, IEEE* **32**(1), 12–19. DOI 10.1109/MCG.2012.24. ISSN 0272-1716.
- MacEachren, A., Roth, R., O’Brien, J., Li, B., Swingley, D. & Gahegan, M. (2012), ‘Visual semiotics & uncertainty visualization: An empirical study’, *Visualization and Computer Graphics, IEEE Transactions on* **18**(12), 2496–2505. DOI 10.1109/TVCG.2012.279. ISSN 1077-2626.
- Mackinlay, J. (1986), ‘Automating the design of graphical presentations of relational information’, *Acm Transactions On Graphics (Tog)* **5**(2), 110–141.
- Mackinlay, J. D., Hanrahan, P. & Stolte, C. (2007), ‘Show me: Automatic presentation for visual analysis’, *Visualization and Computer Graphics, IEEE Transactions on* **13**(6), 1137–1144.
- McCloud, S. (1993), *Understanding Comics*, Kitchen Sink Press. URL <http://books.google.com.br/books?id=5aQNAQAAMAAJ>.
- Mennis, J. & Guo, D. (2009), ‘Spatial data mining and geographic knowledge discovery – an introduction’, *Computers, Environment and Urban Systems* **33**(6), 403–408. DOI 10.1016/j.compenvurbsys.2009.11.001. ISSN 0198-9715. URL <http://www.sciencedirect.com/science/article/pii/S0198971509000817>, Spatial Data Mining–Methods and Applications.
- Moody, D. (2009), ‘The “physics” of notations: Toward a scientific basis for constructing visual notations in software engineering’, *IEEE Trans. Softw. Eng.* **35**(6), 756–779. DOI 10.1109/TSE.2009.67. ISSN 0098-5589.
- Moody, D. & Hillegersberg, J. (2009), Evaluating the visual syntax of UML: An analysis of the cognitive effectiveness of the UML family of diagrams, in D. Gašević, R. Lämmel & E. Wyk, eds, ‘Software Language Engineering: First International Conference, SLE 2008, Toulouse, France, September 29-30, 2008. Revised Selected Papers’, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 16–34.

- Moody, D. L., Heymans, P. & Matulevičius, R. (2010), 'Visual syntax does matter: Improving the cognitive effectiveness of the i* visual notation', *Requirements Engineering* **15**(2), 141–175. DOI 10.1007/s00766-010-0100-1. ISSN 1432-010X.
- Nielsen, J. (1993), *Usability Engineering*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- Norman, D. (1986), Cognitive engineering, in D. Norman & S. Draper, eds, 'User Centered System Design: New Perspectives on Human-computer Interaction', Taylor & Francis, pp. 31–61.
- Oliveira, I., Segura, V., Nery, M., Mantripragada, K., Ramirez, J. P. & Cerqueira, R. (2014), WISE: A web environment for visualization and insights on weather data, in 'WVIS - 5th Workshop on Visual Analytics, Information Visualization and Scientific Visualization', SIBGRAPI 2014, pp. 4–7. URL <http://bibliotecadigital.fgv.br/dspace/bitstream/handle/10438/11954/WVIS-SIBGRAPI-2014.pdf?sequence=1>.
- Paganelli, L. & Paternò, F. (2002), Intelligent analysis of user interactions with web applications, in 'Proceedings of the 7th International Conference on Intelligent User Interfaces', IUI '02, ACM, New York, NY, USA, pp. 111–118. DOI 10.1145/502716.502735. URL <http://doi.acm.org/10.1145/502716.502735>.
- Petre, M. (2013), Reflections on representations: Cognitive dimensions analysis of whiteboard design notations, in M. Petre & A. Van Der Hoek, eds, 'Software Designers in Action: A Human-Centric Look at Design Work', Chapman & Hall/CRC Innovations in Software Engineering and Software Development Series, CRC Press. URL <https://books.google.com.br/books?id=gTDSBQAAQBAJ>.
- Pinker, S. (1990), 'A theory of graph comprehension', *Artificial intelligence and the future of testing* pp. 73–126.
- Quesenbery, W. & Brooks, K. (2010), *Storytelling for User Experience - Crafting Stories for Better Design*, 1st ed., Rosenfeld Media, New York, NY, USA.
- Santos, E., Lins, L., Ahrens, J. P., Freire, J. & Silva, C. T. (2009), 'Vismashup: Streamlining the creation of custom visualization applications', *Visualization and Computer Graphics, IEEE Transactions on* **15**(6), 1539–1546.

- Satyanarayan, A. & Heer, J. (2014), 'Authoring narrative visualizations with ellipsis', *Comput. Graph. Forum* **33**(3), 361–370. DOI 10.1111/cgf.12392. ISSN 0167-7055. URL <http://dx.doi.org/10.1111/cgf.12392>.
- Segel, E. & Heer, J. (2010), 'Narrative visualization: Telling stories with data', *Visualization and Computer Graphics, IEEE Transactions on* **16**(6), 1139–1148. DOI 10.1109/TVCG.2010.179. ISSN 1077-2626.
- Segura, V. C. V. B. & Barbosa, S. D. J. (2016), *History Viewer: Displaying User Interaction History in Visual Analytics Applications*, Springer International Publishing, Cham, pp. 223–233.
- Segura, V., Ferreira, J. J., Cerqueira, R. & Barbosa, S. (2016), Uma avaliação analítica de um sistema de visualização do histórico de interação do usuário usando CDN e PoN, in 'IHC 2016 - Full Papers (to appear)', São Paulo - Brazil. URL <http://XXXXX/160111.pdf>.
- Shneiderman, B. (1997), Direct manipulation for comprehensible, predictable and controllable user interfaces, in 'Proceedings of the 2nd International Conference on Intelligent User Interfaces', IUI '97, ACM, New York, NY, USA, pp. 33–39. DOI 10.1145/238218.238281. URL <http://doi.acm.org/10.1145/238218.238281>.
- Silva, C. T., Anderson, E., Santos, E. & Freire, J. (2011), Using VisTrails and provenance for teaching scientific visualization, in 'Computer Graphics Forum', Vol. 30, Wiley Online Library, pp. 75–84.
- Silva, S. F. & Catarci, T. (2000), Visualization of linear time-oriented data: A survey, in 'Proceedings of the First International Conference on Web Information Systems Engineering (WISE'00)-Volume 1 - Volume 1', WISE '00, IEEE Computer Society, Washington, DC, USA, pp. 310–. URL <http://dl.acm.org/citation.cfm?id=882511.885379>.
- Storey, M.-A. D., Čubranić, D. & German, D. M. (2005), On the use of visualization to support awareness of human activities in software development: A survey and a framework, in 'Proceedings of the 2005 ACM Symposium on Software Visualization', SoftVis '05, ACM, New York, NY, USA, pp. 193–202. DOI 10.1145/1056018.1056045.
- Thiago, R., Azevedo, L., da Silva, V., Segura, V., dos Santos, M. & Cerqueira, R. (2016), CATS: Cognitive analytic trail system. URL <http://www.aaai.org/ocs/index.php/WS/AAAIW16/paper/view/12587>.

- Venkatesh, V. & Davis, F. D. (2000), 'A theoretical extension of the technology acceptance model: Four longitudinal field studies', *Management Science* 46(2), 186–204. DOI 10.1287/mnsc.46.2.186.11926. URL <http://dx.doi.org/10.1287/mnsc.46.2.186.11926>.
- Wohlfart, M. & Hauser, H. (2007), Story telling for presentation in volume visualization, *in* 'Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization', EUROVIS'07, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 91–98. DOI 10.2312/VisSym/EuroVis07/091-098. URL <http://dx.doi.org/10.2312/VisSym/EuroVis07/091-098>.
- Wojtkowski, W. & Wojtkowski, W. G. (2002), Storytelling: Its role in information visualization, *in* 'European Systems Science Congress', Citeseer.
- Yoon, Y., Myers, B. A. & Koo, S. (2013), Visualization of fine-grained code change history, *in* '2013 IEEE Symposium on Visual Languages and Human Centric Computing', pp. 119–126. DOI 10.1109/VLHCC.2013.6645254. ISSN 1943-6092.

A

Visual Analytics Definitions

This appendix lists some definitions found in the literature for *visual analytics*. It is by no means an exhaustive list, just a collection of what we found during the course of this thesis.

- “Visual analytics is the science of analytical reasoning facilitated by interactive visual interfaces.” (Cook & Thomas, 2005)
- “Visual analytics is more than just visualization and can rather be seen as an integrated approach combining visualization, human factors and data analysis. (...) With respect to the field of visualization, visual analytics integrates methodology from information analytics, geospatial analytics, and scientific analytics. Especially human factors (e.g., interaction, cognition, perception, collaboration, presentation, and dissemination) play a key role in the communication between human and computer, as well as in the decision-making process.” (Keim et al., 2006)
- “Visual analytics is the formation of abstract visual metaphors in combination with a human information discourse (usually some form of interaction) that enables detection of the expected and discovery of the unexpected within massive, dynamically changing information spaces. It is an outgrowth of the fields of scientific and information visualization but includes technologies from many other fields, including knowledge management, statistical analysis, cognitive science, decision science, and others.

This marriage of computation, visual representation, and interactive thinking supports intensive analysis. The goal is not only to permit users to detect expected events, such as might be predicted by models, but also to help users discover the unexpected – the surprising anomalies, changes, patterns, and relationships that are then examined and assessed to develop new insight.” (Cook et al., 2007)

- “Visual analytics combines automated analysis techniques with interactive visualisations for an effective understanding, reasoning and decision making on the basis of very large and complex datasets.” (Keim et al., 2010)

- “Visual analytics is an emerging research discipline aiming at making the best possible use of huge information loads in a wide variety of applications by appropriately combining the strengths of intelligent automatic data analysis with the visual perception and analysis capabilities of the human user.” (Kohlhammer et al., 2011)
- “Visual analytics is about creating such working conditions in which humans and computers can utilize their inherent capabilities in the best possible ways while complementing and amplifying the capabilities of the other side.” (Andrienko et al., 2011)
- “The basic idea of Visual Analytics is the integration of the outstanding capabilities of humans in terms of visual information exploration and the enormous processing power of computers to form a powerful knowledge discovery environment.” (Aigner et al., 2007a)

B PROV-XML Example

In this appendix we provide examples for the PROV-XML elements that are being saved with our log model. Section B.1 presents the elements from the *definition hierarchy* whilst section B.2, the ones from the *interaction hierarchy*.

B.1 Definition Hierarchy

The following code excerpts provide examples of the elements of the *definition hierarchy* in PROV-XML format:

- Listing B.1: SrcSysDef
- Listing B.2: ViewDef
- Listing B.3: VisCompDef
- Listing B.4: DataServDef

Listing B.1: Example of source system definition in PROV-XML.

```
<prov:collection prov:id="wisespl:srcSys_WISE-SPL_v0.0.1">
  <prov:label><![CDATA[Weather InSights Environment]]></
    prov:label>
  <prov:type>bonnie:sourceSystem</prov:type>
  <bonnie:shortName>WISE-SPL</bonnie:shortName>
  <bonnie:prefix>wisespl</bonnie:prefix>
  <bonnie:version>0.0.1</bonnie:version>
  <bonnie:webUtils>http://brlwebutils-v0-1.mybluemix.net/min/
    webUtils.js</bonnie:webUtils>
</prov:collection>
```

Listing B.2: Example of view definition in PROV-XML.

```
<prov:collection prov:id="wisespl:viewDef_portal_v0.0.1">
  <prov:label><![CDATA[Portal]]></prov:label>
  <prov:type>bonnie:viewDefinition</prov:type>
  <bonnie:shortName>portal</bonnie:shortName>
  <bonnie:version>0.0.1</bonnie:version>
</prov:collection>
<prov:hadMember>
  <prov:collection prov:ref="wisespl:srcSys_WISE-SPL_v0.0.1"/>
```

```
<prov:entity prov:ref="wisespl:viewDef_portal_v0.0.1"/>
</prov:hadMember>
```

Listing B.3: Example of visualization component definition in PROV-XML.

```
<prov:entity prov:id="wisespl:visComp_portal.map_v0.0.1">
  <prov:label><![CDATA[Map]]></prov:label>
  <prov:type>bonnie:visComponent</prov:type>
  <bonnie:shortName>map</bonnie:shortName>
  <bonnie:version>0.0.1</bonnie:version>
  <bonnie:url>http://localhost:3000/viscomp/MapVisComp.js</
    bonnie:url>
  <bonnie:className>MapVisComp</bonnie:className>
  <bonnie:styles>http://localhost:3000/viscomp/MapVisComp.css</
    bonnie:styles>
  <bonnie:webUtilsDep>mapUtils/mapUtils_arcgis</
    bonnie:webUtilsDep>
</prov:entity>
<prov:hadMember>
  <prov:collection prov:ref="wisespl:viewDef_portal_v0.0.1"/>
  <prov:entity prov:ref="wisespl:visComp_portal.map_v0.0.1"/>
</prov:hadMember>
```

Listing B.4: Example of data service definition in PROV-XML.

```
<prov:entity prov:id="wisespl:dataServDef_serviceInterface_v0
.0.1">
  <prov:label><![CDATA[Service Interface]]></prov:label>
  <prov:type>bonnie:dataServiceDefinition</prov:type>
  <bonnie:shortName>serviceInterface</bonnie:shortName>
  <bonnie:version>0.0.1</bonnie:version>
  <bonnie:url>http://localhost:3000/dataService/
    ServiceInterface.js</bonnie:url>
  <bonnie:className>ServiceInterface</bonnie:className>
</prov:entity>
<prov:hadMember>
  <prov:collection prov:ref="wisespl:srcSys_WISE-SPL_v0.0.1"/>
  <prov:entity prov:ref="
    wisespl:dataServDef_serviceInterface_v0.0.1"/>
</prov:hadMember>
```

B.2 Interaction Definition

The following code excerpts provide examples of the elements of the *interaction hierarchy* in PROV-XML format:

- Listing B.5: Session

- Listing B.6: Window
- Listing B.7: View
- Listing B.8: SrcSysAction
- Listing B.9: DataServCall
- Listing B.10: VisEffect

Listing B.5: Example of session in PROV-XML.

```

<prov:agent prov:id="wisespl:user_example">
  <prov:type>bonnie:user</prov:type>
  <bonnie:localId>example</bonnie:localId>
  <bonnie:globalId>example</bonnie:globalId>
</prov:agent>
<prov:entity prov:id="wisespl:session_0274f094-f084-4350-af68-550797cafbde">
  <prov:type>bonnie:session</prov:type>
  <bonnie:userAgent><![CDATA[Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:48.0) Gecko/20100101 Firefox/48.0]]></bonnie:userAgent>
</prov:entity>
<prov:activity prov:id="wisespl:session_0274f094-f084-4350-af68-550797cafbde_nav">
  <prov:startTime>2016-05-31T16:13:56.347Z</prov:startTime>
  <prov:type>bonnie:sessionNavigation</prov:type>
</prov:activity>
<prov:wasAttributedTo>
  <prov:entity prov:ref="wisespl:session_0274f094-f084-4350-af68-550797cafbde"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAttributedTo>
<prov:wasAssociatedWith>
  <prov:activity prov:ref="wisespl:session_0274f094-f084-4350-af68-550797cafbde_nav"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAssociatedWith>
<prov:wasGeneratedBy>
  <prov:entity prov:ref="wisespl:session_0274f094-f084-4350-af68-550797cafbde"/>
  <prov:activity prov:ref="wisespl:session_0274f094-f084-4350-af68-550797cafbde_nav"/>
  <prov:time>2016-05-31T16:13:56.347Z</prov:time>
</prov:wasGeneratedBy>
<prov:used>
  <prov:activity prov:ref="wisespl:session_0274f094-f084-4350-af68-550797cafbde_nav"/>
  <prov:entity prov:ref="wisespl:srcSys_WISE-SPL_v0.0.1"/>

```

```
<prov:time>2016-05-31T16:13:56.347Z</prov:time>
</prov:used>
```

Listing B.6: Example of window in PROV-XML.

```
<prov:activity prov:id="wisespl:window_559fcd7f-dc94-4c4c
-9256-550797cafc60_nav">
  <prov:startTime>2016-05-31T16:13:56.348Z</prov:startTime>
  <prov:type>bonnie>windowNavigation</prov:type>
</prov:activity>
<prov:entity prov:id="wisespl:window_559fcd7f-dc94-4c4c
-9256-550797cafc60">
  <prov:type>bonnie>window</prov:type>
</prov:entity>
<prov:wasGeneratedBy>
  <prov:entity prov:ref="wisespl:window_559fcd7f-dc94-4c4c
-9256-550797cafc60"/>
  <prov:activity prov:ref="wisespl:window_559fcd7f-dc94-4c4c
-9256-550797cafc60_nav"/>
  <prov:time>2016-05-31T16:13:56.348Z</prov:time>
</prov:wasGeneratedBy>
<prov:used>
  <prov:activity prov:ref="wisespl:window_559fcd7f-dc94-4c4c
-9256-550797cafc60_nav"/>
  <prov:entity prov:ref="wisespl:session_0274f094-f084-4350-
af68-550797cafbde"/>
  <prov:time>2016-05-31T16:13:56.348Z</prov:time>
</prov:used>
<prov:wasAssociatedWith>
  <prov:activity prov:ref="wisespl:window_559fcd7f-dc94-4c4c
-9256-550797cafc60_nav"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAssociatedWith>
<prov:wasAttributedTo>
  <prov:entity prov:ref="wisespl:window_559fcd7f-dc94-4c4c
-9256-550797cafc60"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAttributedTo>
```

Listing B.7: Example of view in PROV-XML.

```
<prov:activity prov:id="wisespl:view_581883c8-23c3-478f-bbc1
-550797cb0704_nav">
  <prov:startTime>2016-05-31T16:13:56.359Z</prov:startTime>
  <prov:type>bonnie>viewNavigation</prov:type>
</prov:activity>
<prov:entity prov:id="wisespl:view_581883c8-23c3-478f-bbc1
-550797cb0704">
```

```

    <prov:type>bonnie:view</prov:type>
    <bonnie:url>http://localhost:3000/portal/portal.ejs</
      bonnie:url>
  </prov:entity>
  <prov:wasGeneratedBy>
    <prov:entity prov:ref="wisespl:view_581883c8-23c3-478f-bbc1
      -550797cb0704"/>
    <prov:activity prov:ref="wisespl:view_581883c8-23c3-478f-bbc1
      -550797cb0704_nav"/>
    <prov:time>2016-05-31T16:13:56.359Z</prov:time>
  </prov:wasGeneratedBy>
  <prov:used>
    <prov:activity prov:ref="wisespl:view_581883c8-23c3-478f-bbc1
      -550797cb0704_nav"/>
    <prov:entity prov:ref="wisespl>window_559fcd7f-dc94-4c4c
      -9256-550797cafc60"/>
    <prov:time>2016-05-31T16:13:56.359Z</prov:time>
  </prov:used>
  <prov:wasAssociatedWith>
    <prov:activity prov:ref="wisespl:view_581883c8-23c3-478f-bbc1
      -550797cb0704_nav"/>
    <prov:agent prov:ref="wisespl:user_example"/>
  </prov:wasAssociatedWith>
  <prov:wasAttributedTo>
    <prov:entity prov:ref="wisespl:view_581883c8-23c3-478f-bbc1
      -550797cb0704"/>
    <prov:agent prov:ref="wisespl:user_example"/>
  </prov:wasAttributedTo>
  <prov:used>
    <prov:activity prov:ref="wisespl:view_581883c8-23c3-478f-bbc1
      -550797cb0704_nav"/>
    <prov:entity prov:ref="wisespl:viewDef_portal_v0.0.1"/>
    <prov:time>2016-05-31T16:13:56.359Z</prov:time>
  </prov:used>

```

Listing B.8: Example of source system action in PROV-XML.

```

<prov:activity prov:id="wisespl:srcAct_8da56df7-2b4c-4d0e-be7a
  -550797d10d25">
  <prov:startTime>2016-05-31T16:13:57.901Z</prov:startTime>
  <prov:label><![CDATA[Loaded page with forecast [[2015-12-11
    00:00:00]] and property [[Temperature]].]]></prov:label>
  <prov:type>bonnie:srcSysAction</prov:type>
  <bonnie:actionType>Navigation</bonnie:actionType>
</prov:activity>
<prov:used>
  <prov:activity prov:ref="wisespl:srcAct_8da56df7-2b4c-4d0e-
    be7a-550797d10d25"/>

```

```

    <prov:entity prov:ref="wisespl:view_581883c8-23c3-478f-bbc1
      -550797cb0704"/>
    <prov:time>2016-05-31T16:13:57.901Z</prov:time>
  </prov:used>
  <prov:wasAssociatedWith>
    <prov:activity prov:ref="wisespl:srcAct_8da56df7-2b4c-4d0e-
      be7a-550797d10d25"/>
    <prov:agent prov:ref="wisespl:user_example"/>
  </prov:wasAssociatedWith>

```

Listing B.9: Example of data service call and data in PROV-XML.

```

<prov:activity prov:id="wisespl:dataServCall_94663fea-a305-458b
  -8994-550797d2b8a9">
  <prov:startTime>2016-05-31T16:13:58.328Z</prov:startTime>
  <prov:type>bonnie:dataServiceCall</prov:type>
  <bonnie:config><![CDATA [{"forecastServiceUrl":"http://
    localhost:3000/forecast/","observedServiceUrl":"http://
    localhost:3000/observed/"}]]></bonnie:config>
  <bonnie:call>getForecastGridList</bonnie:call>
  <bonnie:arguments><![CDATA [{"forecastId":42}]]></
    bonnie:arguments>
</prov:activity>
<prov:entity prov:id="wisespl:data_f4853351-3267-4774-871f
  -550797d2b808">
  <prov:type>bonnie:data</prov:type>
</prov:entity>
<prov:wasAssociatedWith>
  <prov:activity prov:ref="wisespl:dataServCall_94663fea-a305
    -458b-8994-550797d2b8a9"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAssociatedWith>
<prov:wasAttributedTo>
  <prov:entity prov:ref="wisespl:data_f4853351-3267-4774-871f
    -550797d2b808"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAttributedTo>
<prov:wasStartedBy>
  <prov:activity prov:ref="wisespl:dataServCall_94663fea-a305
    -458b-8994-550797d2b8a9"/>
  <prov:starter prov:ref="wisespl:srcAct_8da56df7-2b4c-4d0e-
    be7a-550797d10d25"/>
</prov:wasStartedBy>
<prov:wasGeneratedBy>
  <prov:entity prov:ref="wisespl:data_f4853351-3267-4774-871f
    -550797d2b808"/>
  <prov:activity prov:ref="wisespl:dataServCall_94663fea-a305
    -458b-8994-550797d2b8a9"/>

```



```

    <prov:time>2016-05-31T16:13:58.328Z</prov:time>
  </prov:wasGeneratedBy>
  <prov:used>
    <prov:activity prov:ref="wisespl:dataServCall_94663fea-a305
      -458b-8994-550797d2b8a9"/>
    <prov:entity prov:ref="
      wisespl:dataServDef_legacyServiceInterface_v0.0.1"/>
    <prov:time>2016-05-31T16:13:58.328Z</prov:time>
  </prov:used>

```

Listing B.10: Example of visualization effect and state in PROV-XML.

```

<prov:activity prov:id="wisespl:visEff_7761e814-f759-4038-a084
  -550797e92fd5">
  <prov:startTime>2016-05-31T16:14:04.080Z</prov:startTime>
  <prov:type>bonnie:visEffect</prov:type>
  <bonnie:visTask>Encode</bonnie:visTask>
</prov:activity>
<prov:entity prov:id="wisespl:visState_d2b91061-d3d8-4531-8ab7
  -550797e930fe">
  <prov:type>bonnie:visState</prov:type>
  <bonnie:state><![CDATA[{"grid":{"id":1,"name":"9km Grid",
    "swLatitude":-26.942543,"swLongitude":-48.94369999999998,"
    "neLatitude":-21.601543,"neLongitude":-41.21667000000002,"
    "spatialResolution":9000,"timeResolution":3600000},"bounds"
    :{"swLatLng":{"lat":-23.009279927709184,"lng":
    -43.60897083203029},"neLatLng":{"lat":
    -22.873327937085328,"lng":-43.25191516796827}},"mapText":
    "2015-12-10 22:00:00","property":{"id":"T_2M","name":
    "Temperature","defaultRange":[8,42],"thresholds":
    [10,13,16,19,22,25,28,31,34,37,40],"categoriesNames":null
    ,"categoriesColors":["#ddedf5","#82D1F5","#00B0DA","#00
    B2EF","#008ABF","#00649D","#FFE14F","#FFCF01","#FDB813","#
    DD731C","#B8471B","#D9182D"],"unitOfMeasure":"C"}]]></
    bonnie:state>
</prov:entity>
<prov:wasAssociatedWith>
  <prov:activity prov:ref="wisespl:visEff_7761e814-f759-4038-
    a084-550797e92fd5"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAssociatedWith>
<prov:wasAttributedTo>
  <prov:entity prov:ref="wisespl:visState_d2b91061-d3d8-4531-8
    ab7-550797e930fe"/>
  <prov:agent prov:ref="wisespl:user_example"/>
</prov:wasAttributedTo>
<prov:wasStartedBy>

```

```

    <prov:activity prov:ref="wisespl:visEff_7761e814-f759-4038-
      a084-550797e92fd5"/>
    <prov:starter prov:ref="wisespl:srcAct_8da56df7-2b4c-4d0e-
      be7a-550797d10d25"/>
  </prov:wasStartedBy>
  <prov:wasGeneratedBy>
    <prov:entity prov:ref="wisespl:visState_d2b91061-d3d8-4531-8
      ab7-550797e930fe"/>
    <prov:activity prov:ref="wisespl:visEff_7761e814-f759-4038-
      a084-550797e92fd5"/>
    <prov:time>2016-05-31T16:14:04.080Z</prov:time>
  </prov:wasGeneratedBy>
  <prov:used>
    <prov:activity prov:ref="wisespl:visEff_7761e814-f759-4038-
      a084-550797e92fd5"/>
    <prov:entity prov:ref="wisespl:visComp_portal.map_v0.0.1"/>
    <prov:time>2016-05-31T16:14:04.080Z</prov:time>
  </prov:used>
  <prov:hadPrimarySource>
    <prov:generatedEntity prov:ref="wisespl:visState_d2b91061-
      d3d8-4531-8ab7-550797e930fe"/>
    <prov:usedEntity prov:ref="wisespl:data_ac11cf88-a509-4df7-
      a268-550797d3afa6"/>
    <prov:label>stations</prov:label>
  </prov:hadPrimarySource>
  <prov:hadPrimarySource>
    <prov:generatedEntity prov:ref="wisespl:visState_d2b91061-
      d3d8-4531-8ab7-550797e930fe"/>
    <prov:usedEntity prov:ref="wisespl:data_f9a2039d-6cb0-469c-
      a8af-550797d3a6ce"/>
    <prov:label>cells</prov:label>
  </prov:hadPrimarySource>
  <prov:hadPrimarySource>
    <prov:generatedEntity prov:ref="wisespl:visState_d2b91061-
      d3d8-4531-8ab7-550797e930fe"/>
    <prov:usedEntity prov:ref="wisespl:data_6ead6e7d-8945-4163-85
      b1-550797e2fd82"/>
    <prov:label>forecast</prov:label>
  </prov:hadPrimarySource>
  <prov:hadPrimarySource>
    <prov:generatedEntity prov:ref="wisespl:visState_d2b91061-
      d3d8-4531-8ab7-550797e930fe"/>
    <prov:usedEntity prov:ref="wisespl:data_085b52e0-23d4-4892-8
      d16-550797e301c9"/>
    <prov:label>observed</prov:label>
  </prov:hadPrimarySource>

```

C Definitions Registration Example

Listing C.1: Example of the expected input object for registering the definition hierarchy of a SrcSys.

```
{
  shortName: 'WISE-SPL',
  name: 'Weather InSights Environment',
  version: '0.0.1',
  prefix: 'wisespl',
  webUtils: 'http://brlwebutils-v0-1.mybluemix.net/min/webUtils
    .js',

  viewDefinitions: {
    portal: {
      shortName: 'portal',
      name: 'Portal',
      version: '0.0.1',

      visComponents: {
        map: {
          shortName: 'map',
          name: 'Map',
          version: '0.0.1',
          url: 'http://brlwisesplhs.mybluemix.net/viscomp/
            MapVisComp/MapVisComp.js',
          className: 'MapVisComp',
          styles: [
            'http://brlwisesplhs.mybluemix.net/viscomp/
              MapVisComp/MapVisComp.css'
          ],
          webUtilsDep: [
            'mapUtils/mapUtils_arcgis'
          ]
        },
      },
    },
    profiles: {
      shortName: 'profiles',
      name: 'Profiles',
      version: '0.0.1',
      url: 'http://brlwisesplhs.mybluemix.net/viscomp/
        ProfilesVisComp/ProfilesVisComp.js',
    }
  }
}
```

```

        className: 'ProfilesVisComp',
        styles: [
          'http://brlwisesplhs.mybluemix.net/viscomp/
            ProfilesVisComp/ProfilesVisComp.css'
        ],
        webUtilsDep: [
          'd3Chart/eventProfile',
          'd3ChartPlugin/reference'
        ]
      },
      meteograms: {
        shortName: 'meteograms',
        name: 'Meteograms',
        version: '0.0.1',
        url: 'http://brlwisesplhs.mybluemix.net/viscomp/
          MeteogramsVisComp/MeteogramsVisComp.js',
        className: 'MeteogramsVisComp',
        styles: [
          'http://brlwisesplhs.mybluemix.net/viscomp/
            MeteogramsVisComp/MeteogramsVisComp.css'
        ],
        webUtilsDep: [
          'd3Chart/lineChart',
          'd3ChartPlugin/reference'
        ]
      }
    }
  },
  dataServiceDefinitions: {
    serviceInterface: {
      shortName: 'serviceInterface',
      name: 'Service Interface',
      version: '0.0.1',
      url: 'http://brlwisesplhs.mybluemix.net/dataService/
        ServiceInterface.js',
      className: 'ServiceInterface'
    },
    legacyServiceInterface: {
      shortName: 'legacyServiceInterface',
      name: 'Legacy Service Interface',
      version: '0.0.1',
      url: 'http://brlwisesplhs.mybluemix.net/dataService/
        LegacyServiceInterface.js',
      className: 'LegacyServiceInterface'
    }
  }
}

```

```

}
}

```

Listing C.2: Example of the returned ID tree after registering the definition hierarchy of a SrcSys.

```

{
  "prefix": "wisespl",
  "srcSysId": "wisespl:srcSys_WISE-SPL_v0.0.1",
  "viewDefs": {
    "portal": {
      "viewDefId": "wisespl:viewDef_portal_v0.0.1",
      "visComps": {
        "map": {
          "visCompId": "wisespl:visComp_portal.map_v0.0.1",
          "name": "Map"
        },
        "profiles": {
          "visCompId": "wisespl:visComp_portal.profiles_v0.0.1",
          "name": "Profiles"
        },
        "meteograms": {
          "visCompId": "wisespl:visComp_portal.meteograms_v0.0.1",
          "name": "Meteograms"
        }
      }
    }
  },
  "dataServDefs": {
    "serviceInterface": {
      "dataServDefId": "wisespl:dataServDef_serviceInterface_v0.0.1"
    },
    "legacyServiceInterface": {
      "dataServDefId": "wisespl:dataServDef_legacyServiceInterface_v0.0.1"
    }
  }
}

```

D History Visualization Analytical Study in Detail

As stated in section 6.1, we performed an analytical study of the history visualization notation using the Physics of Notation (PoN) (Moody, 2009) and the Cognitive Dimensions of Notation (CDN) framework (Green & Petre, 1996). For this study, the history visualization notation can be seen in figure D.1. In the image, each row is given a code inside a rectangle, which will be used throughout the evaluation report using a special format (*e.g.*, A.1).

We considered the following interaction scenario with the SrcSys:

A user navigates to the WISE portal, which presents the most recently generated forecast for the next 48 hours (forecast generated at *2015-12-11*) and the first available timestep selected (*2015-12-11T00:00:00Z*). Then, he checks the temperature forecast for a few hours later (*2015-12-11T14:00:00.000Z*). He has the idea of comparing the same hour in a previously generated forecast, so he opens another window and navigates to the WISE portal again, which presents the same initial configuration (forecast generated at *2015-12-11* with the *2015-12-11T00:00:00Z* timestep selected). He selects an older forecast (generated 24 hours earlier, at *2015-*

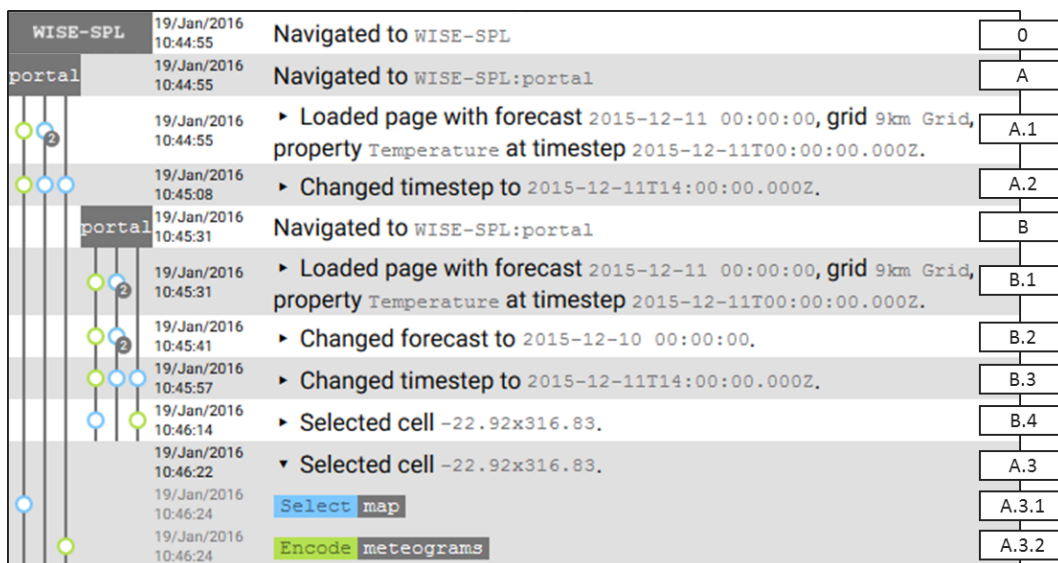


Figure D.1: History visualization iteration for the analytical study.

12-10) and, after loading it, selects the desired timestep (2015-12-11T14:00:00.000Z). Then he chooses to dig further and selects the same cell, with latitude -22.92 and longitude 316.83, on both forecasts. He notices a relevant difference that leads him to an insight about that particular area. On a later day, he needs to trace back the steps that led him to that insight and, therefore, goes to history visualization.

Each analysis was performed by a different evaluator. A developer of the notation was responsible for the PoN analysis and his findings are described in section D.1. Another evaluator, with no familiarity with the developed notation, was responsible for performing the CDN evaluation. Her findings are documented in section D.2. Finally, the last section (section D.3) summarizes the main impacts this study had in the development of the visual notation.

D.1

Evaluation Using Physics of Notation

In the next sections, we will go through each one of PoN's nine principles. Each section opens with a direct quotation from Moodys's work (2009) defining the principle. Then, we proceed to discuss our notation focusing on the principle, highlighting possible trade-offs with other principles as needed.

D.1.1

Principle of Semiotic Clarity

"There should be a 1:1 correspondence between semantic constructs and graphical symbols." (Moody, 2009)

Our visual notation maps our log model onto different representations, as summarized in table D.1. In the table, "n/a" stands for "not available". The "textual" column indicates that some additional text appears alongside the graphical representation, containing a timestamp and a description. The "standard" text does not vary according to the SrcSys (as rows [A], [B], [A.n.m], and [B.n.m]), whilst the "contextual" one is defined by the SrcSys development team and uses familiar terms (as rows [A.n] and [B.n]).

From the table, we can notice a *symbol overload* issue: **Session**s and **View**s are represented using the same graphical construct (gray box with white text). This could be fixed by adding another visual construct to differentiate those semantic constructs. For example, we could change the **Session** row, displaying it graphically as a cell with a black top border (representing the end of the last flow and the beginning of a new one), as shown in figure D.2.

Table D.1: Mapping from log model to our visual notation.

Log Model	Visual	Textual
Session + SrcSysDef	gray box with white text	standard
Window	n/a	n/a
View + ViewDef	gray box with white text	standard
VisCompDef	vertical gray lines	n/a
SrcSysAction	n/a	contextual
DataServiceCall + Data	n/a	n/a
VisEffect + State	colored circles	standard

WISE-SPL	19/Jan/2016 10:44:55	Navigated to WISE-SPL
portal	19/Jan/2016 10:44:55	Navigated to WISE-SPL:portal

Figure D.2: Alternate representation for session.

Several elements of the log model do not have a visual counterpart. This was a conscious design choice by the development team to optimize space. `Window`s are not represented, since we compact the interaction sequences in as few columns as possible. `SrcSysAction`s only have a textual description. Since we wanted to focus on the effects of the `SrcSysAction` in the *visualization components* (the `VisEffect`s), we decided to take the expand/collapse approach and only have the visual representation for the `VisEffect`s. `DataServiceCall`s are not represented because the user is not aware of when they happen (*i.e.*, it is a developer’s choice rather than a user’s choice, and it does not have a direct *visualization effect* necessarily).

D.1.2

Principle of Perceptual Discriminability

“Different symbols should be clearly distinguishable from each other.” (Moody, 2009)

Since we are using different symbols (boxes, lines, and circles) for the different semantic constructs, they are *visually distant* from each other. However, we could still detect two main issues related to this principle: the discriminability between *source system* and *visualization effects*, and the discriminability amongst different *visualization tasks*.

While collapsed, `SrcSysAction` rows may look very similar to the `VisEffect` rows, since they both include colored circles in the graphical representation. We made a design choice to enhance the discriminability between them by adding a collapse/expand symbol near the action textual description. Another design choice in this direction was formatting the `VisEffect` description differently, by using a representation that resembles a “tag” and uses the

same color code as the graphical representation (such as ‘Select’ and ‘Encode’, in figure D.1).

VisEffect nodes are represented by circles with different colors according to the associated *visualization task*. Due to the number of possible *visualization tasks* (11 in total), this difference may not be easily discriminated due to the number of perceptible steps of the color visual variable (around 7-10) (Moody, 2009). One possible solution would be to use different symbols according to the task “macro-category”. This could result in a *perceptual popout* issue, since the different visual variables (shape and color) would be used in combination.

It is important to notice another design choice related to *perceptual discriminability*. In the textual descriptions, we highlight the variable parameters by using a monospaced font, different from the remainder of the text. This way, the user can distinguish between the repeated parts of the description and the variable parameters.

D.1.3

Principle of Semantic Transparency

“Use visual representations whose appearance suggests their meaning.” (Moody, 2009)

We could say our choice of symbols is *semantically opaque*, since it has an arbitrary relationship between appearance and meaning. We can improve their *perceptual resemblance* by using icons for the *visualization effect nodes*, for example. This would also improve the already discussed issue regarding *perceptual discriminability*.

When analyzing the semantic transparency of relationships, we have different transparency levels. On the one hand, we can say that the time relation is *translucent*, since we have a strong Y-axis time constrain and we represent navigational events as breaks on this axis.

On the other hand, the hierarchical structure of the semantic concepts is not so transparent in all cases. The **Window** abstraction is not explicitly represented due to space constraints and to reduce complexity. The log model’s “**Session** contains multiple **Window**s” relationship, therefore, is mapped onto “**Session** contains multiple **View**s” in the notation. This second relationship is expressed by the **Session** representation having a column span according to the number of different simultaneous **View**s, which may be seen as transparent, yet not exactly equal to the underlying semantic relationship.

The log model’s “**ViewDef** contains multiple **VisCompDef**” relationship is transparent by the number of lines coming out from a view. The log model’s “**View** contains multiple **SrcSysAction**s” relationship, however, is more opaque.

The length of a `View`'s *visualization components* lines are not directly related with the number of `SrcSysAction`s that happened in that view. If we have multiple `View`s at the same time, the length of the lines is related to the number of `SrcSysAction`s happening at that time interval. For example, the `View` on the left of figure D.1 has three *actions* (`A.1`, `A.2`, and `A.3`), whilst the one on the right has four (`B.1`, `B.2`, `B.3`, and `B.4`). The left column, however, is longer, since time-wise the `SrcSysAction`s of the `View` on the right happened while the one on the left was still active.

Lastly, the “`SrcSysAction` contains multiple `VisEffect`s” relationship is represented by a collapsed/expanded symbol in the `SrcSysAction`'s description and reinforced by the group sharing the same background color when expanded. This conveys the idea of *spatial enclosure*, making this semantic relationship more transparent.

D.1.4

Principle of Complexity Management

“Include explicit mechanisms for dealing with complexity.” (Moody, 2009)

We are aware of scalability limitations of our representation. A first attempt to reduce the complexity was to group `VisEffect` by `SrcSysAction` and to start the representation with the `SrcSysAction`s collapsed.

When there are too many parallel *views* or too many `SrcSysAction`s, however, complexity problems emerge. We plan to tackle this issue with interactive mechanisms, by allowing to hide/filter/collapse some `Session`s, `View`s, and/or `SrcSysAction`s. This will allow users to explore the visualization at different abstraction levels. Another possibility is to allow “popping out” part of the representation, allowing users to explore the interaction history in a modularized fashion (*e.g.*, for only a given *session*).

Moreover, we plan to experiment with a more compact representation, so more columns could be viewed at any given time. In this compact representation, we can bring the *visualization components* lines together and hide the `VisEffect`s nodes, coloring the lines according to the *visualization task*.

D.1.5

Principle of Cognitive Integration

“Include explicit mechanisms to support integration of information from different diagrams.” (Moody, 2009)

This principle only applies when multiple diagrams are used. Since we are using a single diagram, this principle is not applicable. If we implement



Figure D.3: Session and page context as user scrolls through the visualization.

the aforementioned modularization strategy, this principle will become a major concern.

We can interpret this principle more broadly by considering that, for longer diagrams, only part of them will be visible on the screen at any given time. To contextualize the representation exploration, we keep the current **Session** and **View** on the top rows of the diagram as the user scrolls (as shown in figure D.3). The user, therefore, even without viewing the past **SrcSysAction**s, can *perceptually integrate* the current **SrcSysAction**s with the past **Session**s and **View**s.

D.1.6 Principle of Visual Expressiveness

“Use the full range and capacities of visual variables.” (Moody, 2009)

This principle relates to the number of visual variables (shown in table D.2) being used by the notation. Currently, the following visual variables are carrying information in our notation:

- horizontal position (x): expresses the **VisCompDef** and parallel **Session**s/**View**s;
- vertical position (y): expresses time;
- color: differentiates the *visualization tasks* (node colors) and the **SrcSysAction** grouping (row background color);
- shape: differentiates **Session**/**View** (boxes), **VisCompDef** (lines), and **VisEffect** (nodes).

From the eight available visual variables, our notation currently uses four. We, therefore, have four degrees of visual freedom. We could use the free variables to encode new information.

For example, we could use brightness to represent different, inactive, or collapsed **View**s. In the first case, we could use alternating brightness in different columns, making it easier to perceive the column. Another option is, for a given row, to use a faded out color for the **VisCompDef** lines of the **View**s

Table D.2: Visual variables, their power (highest level of measurement that can be encoded), and capacity (number of perceptible steps).

Variable	Power	Capacity
Horizontal position (x)	Interval	10-15
Vertical position (y)	Interval	10-15
Size	Interval	20
Brightness	Ordinal	6-7
Color	Nominal	7-10
Texture	Nominal	2-5
Shape	Nominal	Unlimited
Orientation	Nominal	4

Source: (Moody, 2009)

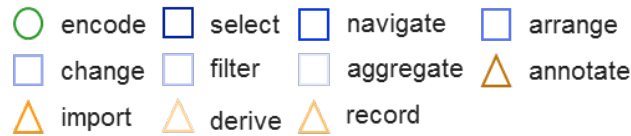


Figure D.4: Example of using different symbols for the different visualization tasks.

unrelated to the row’s `SrcSysAction` or `VisEffect`. Finally, a third scenario would be to use it when a `View` is collapsed (as discussed in the principle of *complexity management*), keeping a faded representation of the collapsed `View`.

Another revision we could make in the light of this principle is the representation of the `VisEffect` node. Instead of using the same shape (circle), and varying the color according to the *visualization task*, we could use another encoding, since color should not be the only discerning visual variable (Moody, 2009). One approach (shown in figure D.4) could be to use color and shape to code the *visualization task* “macro-category” (encode, manipulate, and introduce) and use brightness or texture to distinguish each task type (*e.g.*, annotate, import, derive, and record within the “introduce” category).

D.1.7 Principle of Dual Coding

“Use text to complement graphics.” (Moody, 2009)

We use text across the representation in multiple ways. We provide a more detailed description and the timestamp for each row. Inside the `Session` and `View` symbols, we have the name of the corresponding `Session` and `View`, making it a hybrid (graphic + text) symbol. Every `VisCompDef` line also has a mouseover text to explain its meaning.

Another example of a hybrid symbol used in our representation is the tag-like description used in `VisEffect`s. By using the same *visualization tasks* colors, we provide an integrated “legend” for the `VisEffect` nodes. This way, the user may discover the color meanings by exploration, instead of having to consult another piece of information.

D.1.8

Principle of Graphic Economy

“The number of different graphical symbols should be cognitively manageable.” (Moody, 2009)

From table D.1, we can see that our notation does not use an extensive set of graphical symbols (only boxes, lines, and circles). We can foresee being impacted by this issue if we try to have different symbols for each visualization task. An alternative solution to reduce that problem would be to use mnemonic icons, as previously discussed.

D.1.9

Principle of Cognitive Fit

“Use different visual dialects for different tasks and audiences.” (Moody, 2009)

Our notation currently lacks another dialect, since we are focused in a single use case for the time being (single user interacting with a single SrcSys, as shown in figure 3.1). The aforementioned more compact representation may be seen as a form of dialect, but we have not yet envisioned different representations for expert/novice users or according to the representational medium.

However, we have already discussed applying the same notation for different contexts with simple modifications. For example, instead of focusing on displaying the user interaction history (where the time dimension is very important), we could use the same notation to represent and compare different interaction paths to reach the same goal (focus on the sequence of actions).

D.2

Evaluation Using Cognitive Dimensions of Notation

We performed the CDN evaluation using the cognitive dimensions (CDs) presented in table D.2. We considered the aforementioned scenario as our evaluation context. It is important to notice that the history visualization is a representation from WISE’s domain, which is well known by its users.

Table D.3: Cognitive dimensions for the CDN.

Cognitive Dimension	Description
Abstraction Gradient	What are the minimum and maximum levels of abstraction? Can fragments be encapsulated?
Closeness of mapping	What 'programming games' need to be learned?
Consistency	When some of the language has been learnt, how much of the rest can be inferred?
Diffuseness	How many symbols or graphic entities are required to express a meaning?
Error-proneness	Does the design of the notation induce 'careless mistakes'?
Hard mental operations	Are there places where the user needs to resort to fingers or pencilled annotation to keep track of what's happening?
Hidden dependencies	Is every dependency overtly indicated in both directions? Is the indication perceptual or only symbolic?
Premature commitment	Do programmers have to make decisions before they have the information they need?
Progressive evaluation	Can a partially-complete program be executed to obtain feedback on "How am I doing"?
Role-expressiveness	Can the reader see how each component of a program relates to the whole?
Secondary notation	Can programmers use layout, colour, or other cues to convey extra meaning, above and beyond the 'official' semantics of the language?
Viscosity	How much effort is required to perform a single change?
Visibility	Is every part of the code simultaneously visible (assuming a large enough display), or is it at least possible to juxtapose any two parts side-by-side at will? If the code is dispersed, is it at least possible to know in what order to read it?

Adapted from Green & Petre (1996)

In the next sections, we discuss some issues raised by the CDN analysis. Contrary to the PoN analysis, in which we analyzed each principle individually, for the CDN evaluation we focus on the issues raised, not on individual CDs. For each issue, we present a quick summary – which CDs were affected and whether the cognitive characteristic related to the dimension was present or absent – followed by a brief description and potential solutions for the issue.

D.2.1

The History Visualization Concepts Should be Presented/Explained/Available to Users

CDN: Closeness of mapping

Present/Absent: Absent

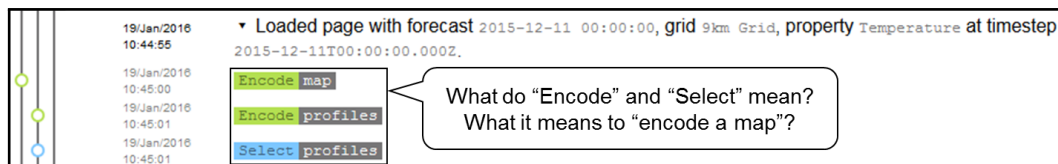


Figure D.5: History visualization concepts not explained to users.

Since BONNIE relies on a new notation, it calls for more explanation. The user is (supposedly) familiar with the SrcSys' concepts from which the history was recorded, but he may not be familiar with the BONNIE's domain. The notation could be introduced to users and have some explanation always available. For example, the first time a user opens the BONNIE, an "explanation view" could appear and he would have the opportunity to go over and learn about the representation.

Additionally, tips on each particular concept of the history representation could be provided once the user hovers the mouse over it. For example, a user would recognize the SrcSys's terminology (*e.g.*, "forecast", "timestep"), but he could have difficulties on relating those concepts with the *visualization tasks* (*e.g.*, "encode", "select", "navigate"), as shown in figure D.5.

CDN: Hidden dependencies & hard mental operations

Present/Absent: Present

Users can only know the color meaning of the nodes once/if he expands an *action row* or hovers the mouse over the node. Since the nodes are colored and the user might need a complete visualization of the history without the details (as provided when BONNIE loads), the color meanings should be clear from the start.

D.2.2

Multiple Nodes Collapsed Into a Single Action Row May Not be What It Seems

CDN: Error-proneness

Present/Absent: Present

Whilst the *action rows* are collapsed, the user can see numbers over some of the nodes. Once he or she hovers the mouse on a node, a tooltip appears indicating the *visualization task* and the *visualization component* related to that given node. For example, in figure D.6, we see the number "2" next to the node and, hovering over the node, a tooltip appears indicating the *visualization task* "select" related to the *visualization component* "profiles".

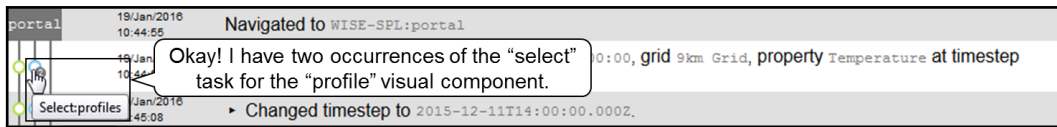


Figure D.6: Multiple nodes collapsed into a single action row.

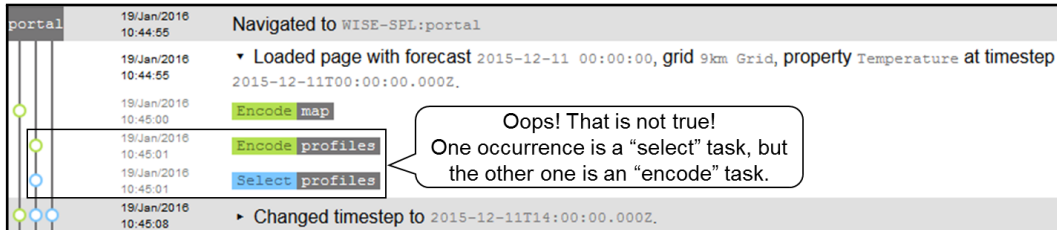


Figure D.7: Expanded action row shows different tasks.

From this representation – and reinforced by the interaction – it seems that there is a given number of the same *visualization task*.

Once the user expands the *action row*, however, he or she sees different *visualization tasks* from those previously inferred. Following with the same example, figure D.7 shows that, after expanding the *action row*, the user does not see two tasks “select”, but one “select” and one “encode”. A node with different colors might help the user understand that more than one type of task is collapsed at that *action row*.

D.2.3

History Visualization Columns Relation to Windows

CDN: Hard mental operations

Present/Absent: Present

The history visualization creates multiple columns – one for each window opened by a SrcSys –, but all columns are equally labeled “portal”. After a while, the user who needs to go over a longer history, with more open windows, may get confused, trying to associate the window with the history column.

D.2.4

Lines Related to Visualization Components

CDN: Hard mental operations & Visibility

Present/Absent: Present & Absent

Each vertical line in the history visualization represents a SrcSys *visualization component*. This representation, however, is not expressed clearly at the interface. The lines for each *visualization component* look the same (color and thickness), giving no opportunity for visual distinction. The user needs to

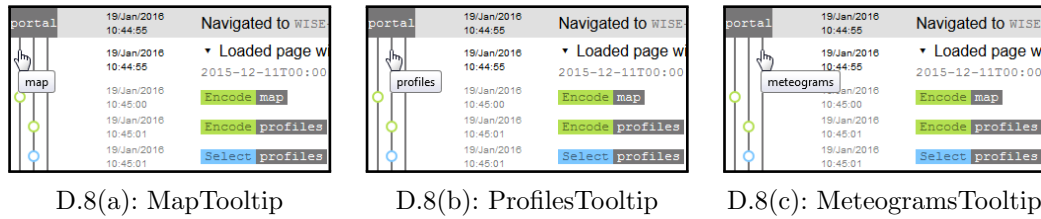


Figure D.8: Each line represents a visualization component.



Figure D.9: User actions versus application actions.

go over each line to check the name of the *visualization component* associated with each row (as shown in figure D.8) or read the descriptions of the *effect rows*.

It may take a lot of cognitive effort to remember which line relates to which *visualization component*, particularly on the collapsed view of the *action row*. When an *action row* is expanded, the descriptions of the child *effect rows* include the corresponding *visualization task* and *visualization component* (e.g., “Encode profiles” in figure D.7). This design choice, however, might harm an overview analysis of the history.

In the scenario, the user focuses on the meteograms of each window. Considering this use case of focusing on a single *visualization component*, the user might want to only see the meteograms’ records. A filter/search feature would be a welcome addition in this case.

Differentiation Between User Actions and Application Actions

- CDN: Diffuseness
- Present/Absent: Present

BONNIE shows actions performed both by the user and by the application, as highlighted in figure D.9. This difference is not easily distinguishable by users, since it relies on interpreting the description of each *action row*. For example, the user might not be completely sure whether he did “encode” something at some point.

This is an issue related to the BONNIE domain notation, defined and created by BONNIE’s designer. The log model should have some way to differentiate actions that were started by an explicit user action (e.g.,

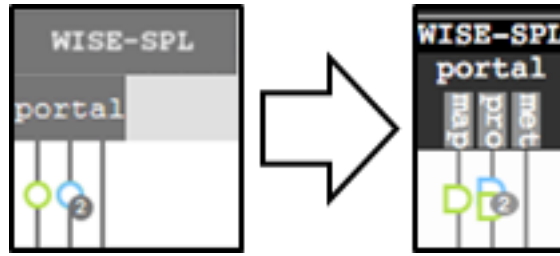


Figure D.10: History visualization analytical study main impacts. Mainly: the change in the navigation row, the cascading nodes, and the different node shapes for different *source system actions*.

navigating to a page) and the ones performed automatically by the SrcSys (e.g., the loading of the default parameters after navigating to a page). In a redesign, the representation should have some visual indicator to distinguish between those two cases.

D.3 Main Impacts

After the analytical study, we made some changes in the visual notation. We considered both evaluators' interpretations and their exploration of the solution space in the redesign of the notation.

We decided to simplify the navigation row by combining the *session* and the *view* navigation. This allowed a more compact representation, since interaction sequences from different *sessions* can appear in the same column. In the visual notation, we created an hierarchical box containing the information from both (thus avoiding the symbol overload issue from PoN's "Principle of Semiotic Clarity"). Additionally, we are also encoding the *visualization components* explicitly in the visual notation, as pointed out by the CDN-raised issue "Lines Related to Visualization Components". Figure D.10 illustrates the difference in the *navigation row* visual notation, showing the original implementation on the left and the updated one on the right.

The CDN-raised issue "Differentiation Between User Actions and Application Actions" reflected a misinterpretation of BONNIE concepts by the evaluator. In this iteration of BONNIE, the `SrcSysAction` was actually called `UserAction` – without sub-types. The evaluator focused on the agent of the action – `UserAction`s being performed by the user and `VisEffect`s was the application's response to the `UserAction`. This led to changes in the documentation and in the log model.

The documentation started to refer to the `VisEffect` as a *visualization impact*. The log model's `UserAction` was generalized to `SrcSysAction`, with four different kinds: user, navigation, system, and BONNIE annotation (as

described in section 4). With these different kinds of `SrcSysAction`, we decided to visualize them with different shapes, a possibility that appeared when analyzing PoN's "Principle of Visual Expressiveness". Figure D.10 shows the different shape for the "navigation" *source system action*.

Finally, the CDN-raised issue "Multiple Nodes Collapsed Into a Single Action Row May Not be What It Seems" was fixed by displaying all nodes when the *action row* is collapsed. The nodes appears in a vertical cascade, allowing to have an idea of both quantity and the task distribution. Figure D.10 illustrates this difference.

E History Visualization User Study Details

Shortly after the analytical evaluation of the history visualization, we conducted a user study. For this study, the version of the history visualization can be seen in figure E.1. There were some changes from the previous study (appendix D), in particular:

- The *navigation rows* were merged, displaying the *source system* and the *view* in the same row, whilst also showing the *visualization components*.
- Added a gray scale to represent the hierarchy from the *source system*, *view*, and *visualization components*. This gray scale is represented both in the representation on the right but also on the colored tags in the rows' textual description.
- When there is more than one *effect node* collapsed, all of them appear overlapping each other, instead of only the last one.
- The column that is not related to a given row appears faded out.

The main objective of the study was to evaluate the proposed history visualization notation, so we did not mention the narrative builder component.

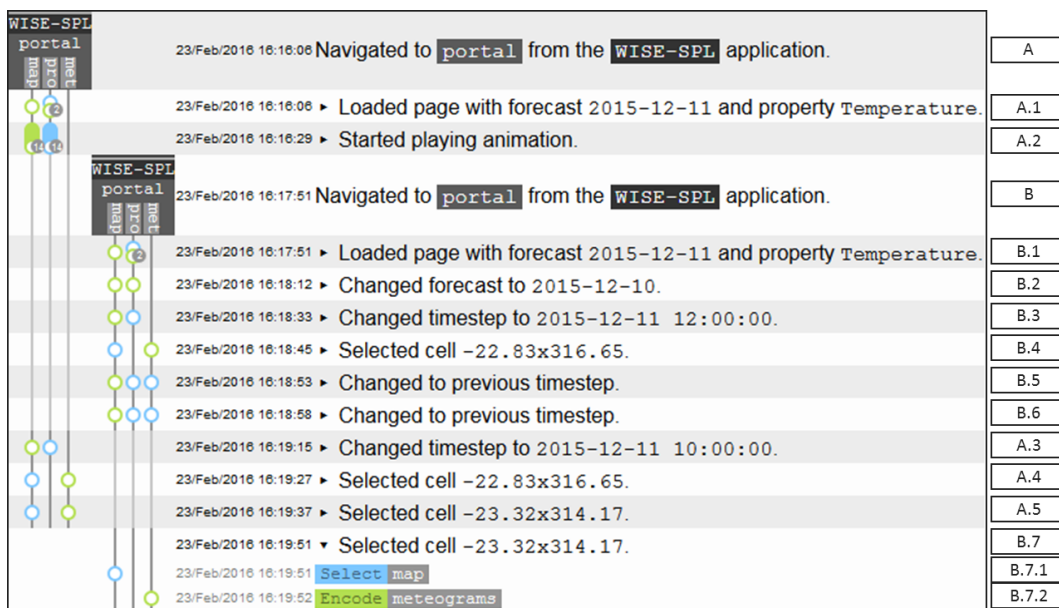


Figure E.1: History visualization iteration for the user study.

Moreover, we also wanted to explore how the participants would map from the SrcSys domain to our BONNIE domain and the impacts of this mapping. Considering the aforementioned interaction sequence (figure 6.10), we devised a series of tasks to study each one of the steps.

The tasks can be summarized as follows:

Task 1: Describe the video The participants should describe a previously recorded interaction video. The idea was to observe the terminology the participants would use to describe the *user actions*.

Task 2: Describe visualization changes The participants should describe the visualization changes for every user action in the previously shown video. Again, the idea was to observe the terminology the participants would use to describe the *visualization effects*. Also, we introduced the idea that a *user action* may trigger several *visualization effects*.

Task 3: Associate visualization tasks The participants should associate a *visualization task* with the visualization changes from task 2. The idea was to introduce the *visualization task* concept.

Task 4: Find the user actions We asked participants to find specific user actions from the video in BONNIE. The idea was to observe if they could relate what happened in the video with the history visualization.

Task 5: Find the visualization states We asked participants to find specific *visualization states* in BONNIE. The idea was to observe if they could relate what happened in the video with the history visualization.

The study was performed with four participants (and an extra pilot study). All participants were professionals, working with software development, and not familiar with BONNIE. The study took about one and a half hour.

The next section (section E.1) contains all the material used in the study. Section E.2 describes the methodology, detailing each task. Section E.3 presents all the collected study data.

E.1 Study Material

This section presents all material used in the study as follows:

- Figures E.2 and E.3 show the initial profile questionnaire.
- Figures E.4 to E.6 present the script used to guide the study session.
- Figure E.7 shows the final questionnaire regarding the history visualization.

1) Are you familiar with any geospatial or temporal geospatial representation?
 Obs.: A geospatial representation displays information related to a location, usually on a map. For example, the traffic information and alerts on Waze. A temporal geospatial representation displays this kind of information over a period of time.
 No (go to question 2)
 Yes

a) Which ones? (mark as many as wanted)
 Google Maps
 Google Earth
 Waze
 Others: _____

b) How often do you use the most frequent one?
 A few times each month
 A few times each week
 Every day
 Many times a day

2) Do you usually check any weather forecast system? (e.g.: weather.com, climatempo, weather underground)
 No (go to question 3)
 Yes

c) Which ones? _____

d) How often (the most frequent one)?
 A few times each month
 A few times each week
 Every day
 Many times a day

e) How do you usually expect to see the information? (mark all that apply, in order of precedence, where 1=the most important one for you)
 As a single number
 As a map representation
 As a chart
 As an icon or symbol
 Other: _____

f) How much time ahead do you usually look for weather information?
 Just for the present time
 Some hours ahead
 Some days ahead
 Some weeks ahead

g) How often do you keep track of the forecast for a given day/time of interest, checking it multiple times to see if it has changed (e.g. for the weekend forecast, check it on Thu evening, Fri morning and Fri evening)?
 Never
 Rarely
 Occasionally
 Frequently

h) Do you usually compare forecasts from different sources (for example, compare the weather.com forecast with the weather underground one)?
 No
 Yes

3) Do you have any previous knowledge regarding WISE (Weather InSights Environment)?
 No, I have never heard of it.
 Yes, I have seen at least one description (presentation, paper, ...) of it, but never used it.
 Yes, and I have already interacted with it.
 Yes, and I consider myself an expert user.

Figure E.2: Profile questionnaire, page 1.

4) Do you use any version control system?
 No (end of the questionnaire)
 Yes

a) For how long have you been using this kind of system?
 Less than a year
 1 – 3 years
 3 – 5 years
 More than 5 years

b) Which ones? (mark as many as wanted)
 CVS
 SVN
 GIT / Mercurial
 Other: _____

c) Which type of user do you consider yourself regarding version control systems?
 A novice user, knowing only the basics.
 A regular user, knowing just enough for my daily activities.
 An expert user, knowing advanced commands and the internal workings of the system.

d) How often do you use version control (the most frequently used system)?
 Never
 Rarely
 Occasionally
 Frequently

e) Do you usually check the commit history?
 Never (end of the questionnaire)
 Rarely
 Occasionally
 Frequently
 Always

f) How often do you use the visual representation of the commit history (example shown below)?

Never, I only use the textual representation (end of the questionnaire)
 Rarely
 Occasionally
 Frequently
 Always

g) What features of the representation help you the most? (mark all that apply)
 The textual descriptions help me understand what the commit was about.
 The horizontal positioning of nodes help me understand in which ramification the commit occurred.
 The vertical positioning of nodes help me perceive the flow of time.
 The colors help me distinguish different tags and ramifications.
 The tags help me scan through the text.
 Other: _____

Figure E.3: Profile questionnaire, page 2.

Test description

- You are going to test a visual representation for the user interaction history of a visual analytics application
- Take your time, there is no right/wrong answers
- If possible, think aloud, just saying whatever comes to your mind, without attempting to create a coherent discourse.

- Agenda:
 - WISE Introduction
 - 2 tasks
 - HistoryViewer Introduction
 - 3 tasks
 - Questionnaire + Interview

WISE Description

- Weather InSights Environment
- Shows weather related information from forecast and observed data

- Each forecast:
 - Is generated **daily at midnight**
 - Comprises **48 hours** of predicted data
 - i.e. a forecast generated on January 1st at midnight predicts data up until January 3rd at midnight
 - Has data for every **1 hour**
 - i.e. the forecasted data timestep is of 1 hour

WISE Description

- Therefore, for a given timestamp there are **2 predicted data**

Forecast

- January 3rd
- January 2nd
- January 1st

Available predicted data

- Jan 1st 00:00
- Jan 2nd 00:00
- Jan 3rd 00:00
- Jan 4th 00:00
- Jan 5th 00:00

For the Jan 2nd 12:00 timestep, there are data from the January 1st and January 2nd forecasts.

For the Jan 3rd 12:00 timestep, there are data from the January 2nd and January 3rd forecasts.

WISE Main UI

- Configuration:** Sets the current visible forecast, timestep, and weather property.
- Map:** Displays forecast (cells / squares) and observed (stations / circles) data for the current configuration.
- Profile:** Shows the rain rate distribution for the current forecast. Also acts as a timeline, allowing the user to choose a timestep.
- Meteograms:** Given a cell, shows how different properties changes through the forecast time span.

Basic scenario

- A user wishes to compare the forecast generated on 11/Dec with the previous one (10/Dec).
- He starts with the latest forecast (11/Dec), watching the animation up until a timestep.
- Then he checks the previous forecast (10/Dec) at the same timestep and some timestep before.
- He proceeds to compare some cells, one in Rio and other in São Paulo, for the same timestep.

Task 1 Describe the video

- You are going to watch a video of a user interacting with WISE.
- In your own words, describe what is happening in the video considering the user's intentions behind the actions.
 - For example (action → achieved intention):
 - Double-clicked on a folder → Opened a folder
 - Double-clicked on a program → Executed the program
 - Clicked on the "bold" button → Made the selected text bold
- Do not worry with the timing of the video: you will be able to play/pause/rewind it at will.

Figure E.4: Study script, page 1.

Task 2

Describe visualization changes

- Using the same video as reference, fill in the table describing the changes in the visual components when a given user action occurs
 - "When I do <user action>, what happens with the <visual component>?"
 - Remember that not every visual component is affected when a user action occurs.
 - Rows in italics denotes actions that do not appear in the video. Based on the other actions, make an estimated guess of what happens.
- Do not worry with the timing of the video: you will be able to play/pause/rewind it at will.

WISE

Visual components actions

	User action	Map	Profile	Meteogram
Configuration	Change forecast	Update data	Update data	Update data*
	Change timestep	Update data	Move highlight	Move highlight*
	Previous / Play / Next	Equal to "change in the timestep"		
Map	Change property	Update data		
	Panning/zooming	Move map		
Profile	Mouse over a cell	Show info		
	Click on a cell	Highlight cell		Update data
Meteogram	Mouse over a profile column		Show info	
	Click on a profile column		Equal to "change in the timestep"	
	Mouse over a meteogram			Show value*
	Click on a meteogram		Equal to "change in the timestep"	

HistoryViewer

Visual representation notation

Navigation to a view is shown as a block in the log, identifying the application, the view, and the visual components.

Each line represents a visual component.

Notes represent, visualization actions.

Visualization actions can happen to different components and/or many times to the same component.

User actions can be expanded in their visualization actions.

Each column groups different user interaction sequences.

Textual descriptions for each view.

Loaded page with forecast 2015-11-11 00:00:00, grid 5m, 0x14, property Temperature at timestep 2015-12-12 12:00:00-00:00:00.

Changed timestep to 2015-12-12 04:00:00-00:00:00.

Loaded page with forecast 2015-11-11 00:00:00, grid 5m, 0x14, property Temperature at timestep 2015-12-12 12:00:00-00:00:00.

Changed forecast to 2015-12-10 00:00:00.

Changed timestep to 2015-12-12 12:04:00-00:00:00.

Selected cell -22.32x316.43.

HistoryViewer

Visualization tasks

Encode	Codify data in the visual representation.
Select	Demarcate one or more elements in the visualization, distinguishing selected from unselected elements (e.g.: select, brush, highlight).
Navigate	Alter user's viewpoint (e.g.: zooming, panning, rotating).
Arrange	Organize visual elements (e.g.: reordering axes, rows/columns).
Change	Alter visual encoding (e.g.: size and transparency of points, changing the chart type).
Filter	Adjust the exclusion and inclusion criteria for elements in the visualization.
Aggregate	Change the granularity of visualization elements.
Annotate	Add graphical or textual annotations associated with one or more visualization elements.
Import	Add new elements to the visualization.
Derive	Compute new data elements given existing data elements.
Record	Save or capture visualization elements as persistent artifacts.

Task 3

Associate visualization tasks

- Fill in the table associating each visual component action with a visualization task.
 - Only one visualization task per action
 - The same visualization task may appear several times
- You may use the video, but it is not necessary.

WISE

Visualization tasks

Visualization Actions	Visualization Task
Update data	Encode
Move highlight	Select
Move map	Navigate
Show [info/value]	Annotate
Highlight cell	Select

Figure E.5: Study script, page 2.

The image shows a study script page with two task boxes. The left box, labeled 'Task 4', contains instructions for finding user actions. The right box, labeled 'Task 5', contains instructions for finding visualization states. Both boxes include bulleted instructions and a small page number in the bottom right corner.

Task 4
Find the user actions

- We will show you some short clips from the video in a random order.
- For each one of the short clips, indicate the corresponding row in HistoryViewer.
- You may replay each short clip, but not the entire video.

13

Task 5
Find the visualizations states

- Consider that you can select a visualization action row (the ones with the colored tags) and get the image of the given visualization at that instant.
- Indicate which rows would you select if you wish to compare the **different** forecasts at the **same** timestep.
- You cannot use the video and/or the short clips.

14

Figure E.6: Study script, page 3.

	1: strongly disagree	2	3	4	5: strongly agree
1. The visual representation notation was easy to understand.					
2. I didn't have to go back to the reference sheet/tutorial to remember the meaning of some elements of the visual representation.					
3. The user interaction history was easier to understand using the visual representation than only reading the descriptions.					
4. It was easy to associate what happened in the video with the representation.					
5. I could remember parts of the video by using only the visual representation.					
In the visual representation:					
6. The hierarchy (application > page > visual components > user actions > visualization actions) was clear.					
7. I noticed the colors for the hierarchy (the gray tones) when reading it.					
8. I could distinguish the order of the visual components in the visual representation (i.e.: map is the leftmost one, profile is the middle one, meteograms are the rightmost one).					
9. It was easy to distinguish between actions that occurred in different windows.					
10. It was easy to distinguish between a user action and a visualization action.					
11. It was easy to notice which visualization actions are part of a given user action.					
12. The different text style for parameters that change in similar user actions (e.g.: "Changed timestep to <parameter>") helped me distinguish between similar user actions.					
13. It was easy to find the user actions in task 4.					
14. When collapsed, it was still easy to notice which visual components were affected by the user action.					
15. When collapsed, it was still easy to interpret which visualization actions happened given a user action.					
16. It was easy to find the visualization actions in task 5.					
17. I noticed the colors for the visualization tasks (the different hues) when reading them.					
18. The similarity with the Git commit graph helped me read the visualization.					

Figure E.7: Final questionnaire.

E.2 Methodology

The study began with an initial questionnaire to gather the participants' profiles (figures E.2 and E.3). It included questions about previous experience and familiarity with similar VApp systems (weather forecast and version control systems) and representations (geospatial – temporal or not – and commit history representations).

After they filled out the initial questionnaire, we used a series of slides to guide the study (figures E.4, E.5, and E.6). We started with basic instructions regarding the study and its agenda (slide 1). We proceeded with a brief introduction to WISE (slide 2). We explained how the data give us two values for a given timestep, since there are two forecasts generated independently for any given timestep (slide 3). Then we described WISE's UI, explaining each *visualization component* (slide 4).

After the introduction to WISE, we briefed the participants on the study scenario, which focused on comparing two different forecasts (slide 5). Instead of interacting directly with WISE, participants interacted with a screen-capture video of another person's interaction with WISE. Every participant, therefore, had access to the same interaction sequence and was not affected by loading time or connection issues. It is important to highlight that, given the main objective to compare two different forecasts, the video was recorded using two windows side-by-side: one for each forecast. This was a design choice to (i) make the comparison easier (since both forecasts were visible side-by-side); (ii) to avoid too much flickering in the video when switching windows, and (iii) to make it easier for participants to discern between forecasts.

Task 1 (slide 6) asked participants to narrate the video using their own words to describe each click in terms of user's intention (*e.g.*, “opened a folder” and not “double-clicked on a folder”). The idea with this task was to focus the participant's attention on the sequence of user actions shown in the video. Since the participant was not interacting with the SrcSys itself, we tried to engage the participant, avoiding a “passive viewer” attitude.

Another objective of this task was to help understand WISE's terminology. It gave us an opportunity to correct participants' discourse if a wrong term was used in their descriptions (*e.g.*, mistake a “timestep” for a “forecast”). In the expected usage of BONNIE, the user will be familiar with the terminology of the SrcSys. Since our focus was on BONNIE, we tried to help participants with WISE's terminology, since it could negatively impact the understanding of the log on the next tasks.

Task 2 (slide 7) asked to describe how each user action impacted

each *visualization component*. The idea was to help participants notice such impacts and reinforce the *visualization component* concept. After this task, we presented our version of answers (slide 8), to promote discussion and establish a common ground for the next task.

After these two tasks, we introduced the current version of the history visualization (slide 9). Using a different interaction sequence, we explained the notation and asked the participants to interact with the system. In this step, they could ask anything about the notation. We also presented the list of *visualization tasks* with short descriptions (slide 10).

Task 3 (slide 11) asked to associate each *visualization effect* (from task 2) with a *visualization task*. As Task 3 depends on Task 2, and because we did not want participants' mistakes on Task 2 to negatively influence their performance in Task 3, we gave participants our version of the correct answers to Task 2 (slide 8) before asking them to perform Task 3. Every participant, therefore, had to associate the same five *visualization effects*.

The aim of this task was to familiarize participants with the representation he finds in BONNIE, still anchoring with what may happen in WISE. After this task, we presented the association that was used by WISE's designer and is actually codified in BONNIE, discussing the differences with the participants (slide 12).

We then showed BONNIE with the interaction sequence of what happened in the video (figure E.1 shows a screenshot of this representation). At this point, we told the participants they would no longer have access to the entire video and they would only interact with BONNIE. This would be akin to what a real user would encounter if he wanted to go over an interaction sequence steps using BONNIE. The user would not have access to a playback of his or her steps, only the displayed log information and what he could remember.

For task 4 (slide 13), we presented five short clips from the video – each one containing a single click from the original video and in a random order – and asked the participants to associate them with the corresponding *action row* from BONNIE. The idea was to see whether the participants were able to identify the correct row in the log representation.

In task 5 (slide 14), we asked participants to choose which *effect rows* corresponded to a given *visualization component* in a given timestep, for each one of the forecasts. The aim of this task was similar to the previous one, but now considering the *effect rows*.

After completing all tasks, we asked the participants to answer a final questionnaire (figure E.7). This final questionnaire contained 18 statements,

each one with a Likert scale going from 1 (strongly disagree) to 5 (strongly agree). Every statement was formulated in such a way that a higher score is better. This final questionnaire covered many topics and acted as a conversation starter to an informal interview at the end.

E.3 Study Results

From the initial questionnaire results (table E.1), we saw that all participants were familiar with some geospatial representation (everyone declared being familiar with Google Maps) and have used it at least a few times every week.

Regarding weather forecast systems, only P1 answered that he does not use any of such systems. The other participants use such system mainly in their smartphones, with two of them (P2 and P4) checking it daily. Most of them expect to see information a few hours ahead. None of them has the habit of comparing forecasts from different sources (which would be similar to our study scenario).

Considering the familiarity with WISE specifically, we got different answers. P4 declared “I have never heard of it.” P2 chose “I have seen at least one description (presentation, paper, ...) of it, but never used it.” P0 and P3 marked “I have already interacted with it.” Finally, P1 picked “I consider myself an expert user.”

Regarding version control systems (VCS), all participants were users of some VCS. Only P2 had been using VCS for less than a year, whilst the others had been using a VCS for more than 3 years. Every participant was a GIT user, with different expertise levels and using it frequently. With the exception of P2, the participants check the commit history, even if rarely. Most participants that check the commit history use the visual representation occasionally (only P1 uses it rarely). The feature from the visual representation most used by the participants is the textual description (used by 4 participants). It is followed by the node positioning in the horizontal and vertical axes to denote the ramification and flow of time respectively (used by 3 participants each). Only P4 declared being aided by the colors and tags of the representation.

Task 1 allowed us to compare how the participants expressed the *source system actions* with how the WISE designer expressed them (results in table E.2). It was interesting to notice the variation of terms amongst participants and even considering the same participant. For example, P1 used both “pick” and “choose” for the “select cell” user action.

Table E.3 summarizes the findings, showing on the left column the

Table E.1: Profile questionnaire results.

Question	P0	P1	P2	P3	P4
1	yes	yes	yes	yes	yes
1.a.	1	1	1	1	1
Google Maps					
1.a.	1	1	0	1	1
Google Earth					
1.a. Waze	1	0	1	1	1
1.a. Other	0	Open Street Maps	0	Bing	0
1.b	3	2	2	3	2
2	yes	no	yes	yes	yes
2.c	Google Now	0	iPhone	Weather Channel	Accu- weather
2.d	2	0	3	1	3
2.e. Single number	4	0	1	0	4
2.e. Chart	1	0	0	0	1
2.e. Map	2	0	0	0	3
2.e. Icon	3	0	0	0	2
2.e. Other	0	0	0	As a list of forecast during the day	0
2.f	1	0	2	2	3
2.g	2	0	1	3	3
2.h	no	0	no	no	no
3	3	4	2	3	1
4	yes	yes	yes	yes	yes
4.a	4	4	1	4	3
4.b. CVS	0	0	0	0	0
4.b. SVN	1	0	0	1	1
4.b. GIT	1	1	1	1	1
4.b. Other	0	0	0	RTC, Visual Source Safe	0
4.c	2	3	1	2	3
4.d	4	4	4	4	4
4.e	4	2	1	2	3
4.f	3	2	1	3	3
4.g. Tex- tual	1	1	0	1	1
4.g. Hori- zontal	0	1	0	1	1
4.g. Ver- tical	0	1	1	1	1
4.g. Colors	0	0	1	0	1
4.g. Tags	0	0	1	0	1
4.g. Other	0	0	0	0	0

Table E.2: Task 1 results.

Item	Our Answer	P0	P1	P2	P3	P4
Load page	Load	Open	Enter	Open	Open	Load
Started playing animation	Start	Animate	Play	Animate	Play	Start
Load page	Load	Open	Enter	Compare	Open	Load
Changed forecast	Change	Choose	Switch	Compare	Choose	Select
Changed timestep	Change	Choose	Put	Select	See	Select
Selected cell	Select	Choose	Choose	See	See	Select
Changed to previous timestep	Change	Back	Back	Back	Back	Back
Changed to previous timestep	Change	Back	Back	Back	Back	Back
Changed timestep	Change	Choose	Put	Select	Choose	Select
Selected cell	Select	Choose	Pick	See	Choose	Select
Selected cell	Select	Choose	Choose	See	Choose	Select
Selected cell	Select	Choose	Choose	See	Choose	Select

Table E.3: Summary of task 1 results.

WISE Designer	Participants' answers
Loaded page	Open (5), Enter (2), Load (2), Compare (1)
Started playing animation	Animate (2), Play (2), Start (1)
Changed forecast	Choose (2), Compare (1), Select (1), Switch (1)
Changed timestep	Select (4), Choose (3), Put (2), See (1)
Changed to previous timestep	Back (10)
Selected cell	Choose (10), See (5), Select (4), Pick (1)

description of the *source system action* by the WISE designer and the verbs used by the participants, organized from most often used to least often used (the number in parenthesis indicates the number of occurrences). It is possible to notice that the WISE designer's choices do not correspond to the most frequent occurrence among participants for every *source system action*. Having a shared vocabulary could make the cognitive workload of interpreting the log easier and should be a WISE designer's concern.

Task 2 had a similar outcome as task 1 (results in table E.4), with many synonyms being used to describe the same effects on *visualization components*, as summarized in table E.5 (which follows a structure similar to table E.3). One interesting observation was that most participants were able to guess how the *visualization components* would behave with *actions* that were not available in the video. For example, the video does not show what happens when the weather property is changed. The participants, however, were able to infer what the effects on the *visualization components* were.

In the first two tasks, the participants could express themselves using their own words. In the third task, however, they were asked to adapt to a

Table E.4: Task 2 results.

Item		Our Answer	P0	P1	P2	P3	P4
Change forecast	map	Update	Change	Load	✓	Load	✓
	prof	Update	Change	Load	✓	Load	✓
	met	Update	Change	Load	Change	x	x
Change timestep	map	Update	Change	Change	Change	Load	✓
	prof	Highlight	Show	Change	Change	✓	Select
	met	Highlight	x	Change	Change	x	x
Change property	map	Update	Change	Change	Change	Change	Show
	prof	x	✓	✓	Change	Present	Change
	met	x	✓	✓	✓	✓	Show
Panning / zooming	map	Move	Pan/zoom	Modify	Show/HidePosition		Update
	prof	x	✓	✓	✓	✓	✓
	met	x	✓	✓	✓	✓	✓
Mouse over a cell	map	Show	Appear	Say	Appear	Present	✓
	prof	x	✓	✓	✓	✓	✓
	met	x	✓	✓	✓	✓	✓
Click on a cell	map	Highlight		Show	x	✓	Show
	prof	x	✓	✓	✓	✓	✓
	met	Update	✓	Change	✓	Show	✓
Mouse over profile	map	x	✓	✓	✓	✓	✓
	prof	Show	Appear	Say	✓	✓	✓
	met	x	✓	✓	✓	✓	✓
Mouse over met.	map	x	✓	✓	✓	✓	✓
	prof	x	✓	✓	✓	✓	✓
	met	Show	Say	Say	x	✓	✓

new model (the 11 *visualization tasks* shown in table 4.1). Many participants hesitated between two or three *visualization tasks*. Table E.6 summarizes the results, displaying the *visualization task* as it was codified in the BONNIE (the “Our Answer” column) and the participants’ answers (a checkmark if equal to ours) and second guesses (inside parenthesis).

The participants’ most common strategy was to use the “change” *visualization task*. Since its description read as “alter visual encoding”, it could be used as an “umbrella” *visualization task*, considering we were only focusing on effects on the *visualization components* – therefore some visual change is supposed to happen.

Moreover, one common discussion issue was the *visualization task* “subject”. Since the *visualization tasks* were expressed using verbs, the participants assumed that the *visualization tasks* were actions performed by the user, and not an effect from an action. For example, the “annotate” *visualization task* was described as “add graphical or textual annotations associated with one or more visualization elements.” Some participants interpreted that the user would explicit add an annotation, not considering when the system added a

Table E.5: Summary of task 2 results.

Study Version	Participants' answers
Update data	Change (12), Update (8), Load (7), Show (2)
Move map	Pan/zoom (1), Modify (1), Show/Hide (1), Position (1), Update (1)
Highlight cell	Show (2), Highlight (1)
Move highlight	Change (4), Highlight (1), Select (1), Show (1)
Show info/value	Show (6), Appear (4), Say (4), Present (1)

Table E.6: Task 3 results.

Item	Our Answer	P0	P1	P2	P3	P4
Update data	Encode	✓	✓ (Change)	Change	Change (Encode)	Change (Derive)
Move highlight	Select	✓	✓	✓	✓ (Navigate)	✓
Move map	Navigate	✓	✓ (Change)	✓ (Aggregate, Change)	✓	✓ (Encode)
Show info/value	Annotate	✓	Encode	✓	Encode (Import)	✓ (Filter)
Highlight cell	Select	✓	✓	Change	✓	✓

tooltip, for instance.

After this study, we decided not to express the *visualization tasks* as a single verb. Rather we decided to make the subject/object clearer. For example, the “annotate” task became “annotated element”.

As previously mentioned, tasks 1, 2, and 3 compared how WISE developer and participants expressed themselves. Tasks 4 and 5 focused on how participants would interpret the history representation and relate to actions in the video.

Task 4 focused on the *source system actions* representation. Table E.7 summarizes the results, showing the expected answer (considering the row codes from figure E.1) and a checkmark if the participant got the correct answer, or the type of mistake that was made otherwise. It is important to notice that, even with the right answers, some items caused uncertainty with participants, as described later when discussing the interview results.

While watching the short clips, many participants uttered their version of the *user action* description and were confident if they found a similar one in the representation. This observation can be combined with the results of task 1, since even without using the same wording, participants were able to grasp the *user action*.

P2 used the graphical representation constantly to confirm his choices.

Table E.7: Task 4 results. The “Answer” column is based on the row codes from figure E.1.

Item	Answer	P0	P1	P2	P3	P4
A	A.3	✓	✓	✓	✓	✓
B	B.2	✓	✓	✓	✓	✓
C	A.5	✓	wrong view	✓	wrong cell	✓
D	A.2	✓	✓	✓	✓	✓
E	B.4	✓	✓	✓	✓	✓

When asked why he had chosen a given row, he explained using the textual description and the nodes. This approach, however, got him in doubt when the short clip showed the “started playing animation” user action (item D) with only two timesteps (to keep the short clip short) and he found a user action with 28 *visualization effects* in the representation (since in the entire video the animation goes through 14 timesteps and impacts two *visualization components*). In this case, even without understanding the amount of *visualization effects*, he used an “exclusion” strategy, since he could not find any other *source system action* he could associate to the animation.

P1 and P3 made a mistake at the same item in task 4 (item C), in which the user selects a cell in one of the windows. Since the same cell (*i.e.*, same latitude and longitude) is selected in both windows, the participants found the *source system action* description repeated, and they were expected to use the graphical representation to discern between windows. P1 later commented that he noticed the error when he figured out the representation, but did not go back to correct himself.

P3 had a different concern, since he did not understand the representation of the latitude and longitude coordinates as “`latXlng`”. He, therefore, ended with 4 different “selected cell” *source system actions* that he was unable to distinguish. This reinforced the need of the SrcSys to express the *source system actions* with a terminology familiar to its users.

Another participant, P4, also had this kind of doubt. He knew the representation was consistent (as in, the interaction sequences were grouped by column), but could not associate which column corresponded to which window. Moreover, since at this point he was focused on one of the last *source system actions*, he did not go back to the beginning of the interaction, when the difference between windows was clearer. Many times during this task, he commented about not knowing which “instance” of WISE was which. In the end, he guessed correctly, but was not confident of his answer.

Task 5 presented more errors, having two items that were not performed correctly by all participants. Table E.8 shows the results, in a similar structure

Table E.8: Task 5 results. The “Answer” column is based on the row codes from figure E.1.

Item	Answer	P0	P1	P2	P3	P4
Met 2nd cell pair	A.5.2	✓	✓	✓	✓	✓
	B.7.2	✓	✓	✓	✓	✓
Met 1st cell pair	A.4.2	✓	✓	✓	✓	✓
	B.6.2	wrong time	wrong time	can't find time	wrong time	wrong time
Profile	A.3.1	✓	✓	✓	✓	✓
	B.6.1	can't find time	✓	can't find time	✓	✓
Map	A.2.28	wrong time	wrong time	can't find time	wrong view	wrong view
	B.3.2	✓	✓	✓	✓	✓

as table E.7.

The first item got wrong by all participants asked them for the meteo-grams for the 1st cell pair at the 10am timestep ([A.3](#)). In one of the windows, the user explicitly selected the 10am timestep. In the other, the user selected the 12pm timestep ([B.3](#)), selected the cell ([B.4](#)), and then changed the timestep to the previous one twice ([B.5](#) and [B.6](#)), going back to the 10am timestep. In task 1, every participant chose the “going back” description for the *source system action*, but they lost that context when analyzing the representation. A possible solution would be to instruct the SrcSys designer to always inform in the *source system action* description the new values of parameters (*e.g.*, “changed to previous timestep (10am)”).

The second item got wrong by all participants asked for the map at the 12pm timestep. In the beginning of the video, the user played the animation up until the 12pm timestep in one of the windows ([A.2](#)), while directly selecting the timestep in the second one ([B.3](#)). Some participants noticed the animation and assumed that the correct timestep would be amongst one of the possible 14 map *visualization effects* from the animation. None of them used the time constraint to deduce that the final state (12pm) would be in the last *visualization effect*.

From analyzing the participants’ strategies, it was easy to notice the focus on textual description and on bits of data within it. For example, in the “meteo-grams for the 1st cell pair at the 10am timestep” item, they focused on the “10am” bit of information and searched for it in textual descriptions. Not finding it, they ignored it and only focused on the “selected cell” action or gave up.

Another common mistake was not using the graphical representation. For example, P3 selected a *visualization effect* from the wrong window. After the

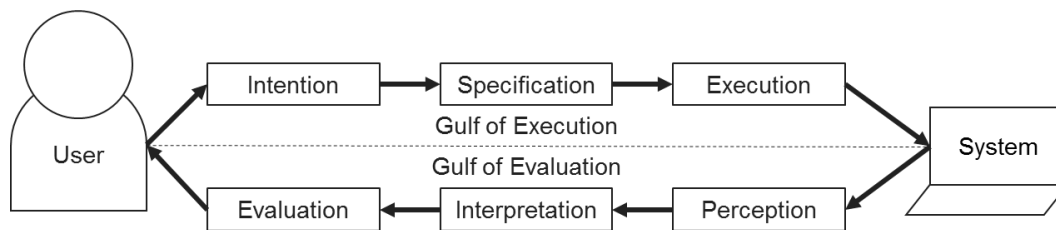


Figure E.8: Norman's seven stages of action.

Adapted from (Norman, 1986)

test, when asked to focus on the graphical representation, he instantly noticed the distinction regarding windows, stating “now it is a piece of cake.”

This difficulty of mapping the video onto the representation may be explained by the distance between the execution and evaluation gulfs (Norman, 1986), as illustrated in figure E.8. If we consider the communication sequence expressed in figure 6.10, it is easy to notice the execution and evaluation gulfs when interacting with the SrcSys (2.a and 2.b) and with BONNIE (6.a and 6.b). We can consider a more abstract gulf, encompassing the execution gulf with the SrcSys (2.a) and BONNIE evaluation gulf (6.a). This is due to the nature of BONNIE, since the data it encodes comes from the interaction with SrcSys.

This distance between the execution in the SrcSys and the evaluation in BONNIE might explain the lack of context experienced by many participants. In fact, when asked whether they remembered the video, all participants answered negatively. When asked to see whether the representation helped remembering, they made the association.

We could argue that the fact we used a video instead of interacting with WISE may have impacted the study negatively, since the participants were more prone to forget the steps, as they had not interacted with the SrcSys themselves. We may also argue that, since it was an “artificial” scenario, the participants did not have the motivation to accomplish the goals underlying the steps. This argument is reinforced by the initial questionnaire answers, in which participants stated that they do not usually compare forecast from different sources.

Either way, both considerations point towards having a mechanism to allow the user register his intentions/interpretation/annotation to be later presented in the BONNIE. This would allow the user to directly interfere with the log visualization, instead of relying only in the description provided by the SrcSys designer and their own memory about the sequence of steps performed in the SrcSys. Ultimately, this consideration led to implement the *BONNIE annotation action* feature.

The final questionnaire answers can be seen in table E.9. Figure E.9 shows the distribution of the answers.

Only two statements had more disagreeing answers than agreeing ones: statements 7 and 15. Statements 4, 14, and 16 leaned towards neutrality. Statements 5 and 17 had mixed feelings. All remaining statements had more agreeing answers.

Regarding the most disagreeing answers, statement 7 related to the gray tones for expressing the hierarchy among BONNIE concepts. When asked, most participants did not even notice the gray tones, resulting in the disagreeing answers. Participants, however, did not seem negatively affected by this design decision, since statement 6 (“the hierarchy was clear”) was agreed by all participants.

The other statement with most disagreeing answers (statement 15) asked about identifying the *visualization effect* given only the *source system action* information. To extract this information, the user should notice the color and location of the nodes. The color would indicate the *visualization task*, and the location would indicate the *visualization component*. Since noticing the colors was one of the mixed feeling questions (statement 17) and interpreting the position was one of the neutral questions (statement 14), the outcome of this question was somewhat expected. Moreover, to take advantage of the color code, the user should be used to the *visualization tasks* and the SrcSys, which was not the case in the study setup.

The other neutral statement was about associating the representation with the video (statement 4) and finding the *visualization effects* in task 5 (statement 16). As previously mentioned, most participants could not remember the video at this point in the study, so the association with the video was somehow affected. Also, as previously discussed, task 5 was the one in which participants had more difficulties, explaining the slightly neutral overall result.

From the most agreed statements, the only one with a “perfect score” (statement 2) was about consulting the printed help material. None of the participants had to go back to the help material during tasks 4 and 5. This can be considered as an indicator of the easiness of the representation, reinforced by the answers to statement 1, which got all answers agreeing with the statement.

Other statements that received agreement from all participants were statements 6, 11, and 13. Statement 6 discussed the BONNIE concepts hierarchy. Participants felt it was easy to grasp the hierarchy, even without considering the gray tones hint as previously mentioned. Statement 11 explored the idea of cause and consequence between *source system action* and *visualization*

Table E.9: Final questionnaire answers. 1 stands for “strongly disagrees” and 5 to “strongly agree”.

Statement	P0	P1	P2	P3	P4
1. The visual representation notation was easy to understand.	4	4	4	5	4
2. I didn't have to go back to the reference sheet/tutorial to remember the meaning of some elements of the visual representation.	5	5	5	5	5
3. The user interaction history was easier to understand using the visual representation than only reading the descriptions.	3	5	5	3	5
4. It was easy to associate what happened in the video with the representation.	3	4	3	3	3
5. I could remember parts of the video by using only the visual representation.	3	4	3	1	5
6. The hierarchy (application > page > visual components > user actions > visualization actions) was clear.	5	4	5	4	5
7. I noticed the colors for the hierarchy (the gray tones) when reading it.	1	5	1	5	2
8. I could distinguish the order of the visual components in the visual representation (i.e.: map is the leftmost one, profile is the middle one, meteograms are the rightmost one).	3	n/a	5	5	5
9. It was easy to distinguish between actions that occurred in different windows.	5	2	5	5	2
10. It was easy to distinguish between a user action and a visualization action.	3	5	5	4	4
11. It was easy to notice which visualization actions are part of a given user action.	5	4	5	5	4
12. The different text style for parameters that change in similar user actions (e.g.: “Changed timestep to <parameter>.”) helped me distinguish between similar user actions.	5	4	5	1	2
13. It was easy to find the user actions in task 4.	5	5	4	4	4
14. When collapsed, it was still easy to notice which visual components were affected by the user action.	3	3	5	4	3
15. When collapsed, it was still easy to interpret which visualization actions happened given a user action.	1	4	1	1	3
16. It was easy to find the visualization actions in task 5.	2	3	3	3	2
17. I noticed the colors for the visualization tasks (the different hues) when reading them.	5	4	1	1	3
18. The similarity with the GIT commit graph helped me read the visualization.	n/a	5	1	4	5

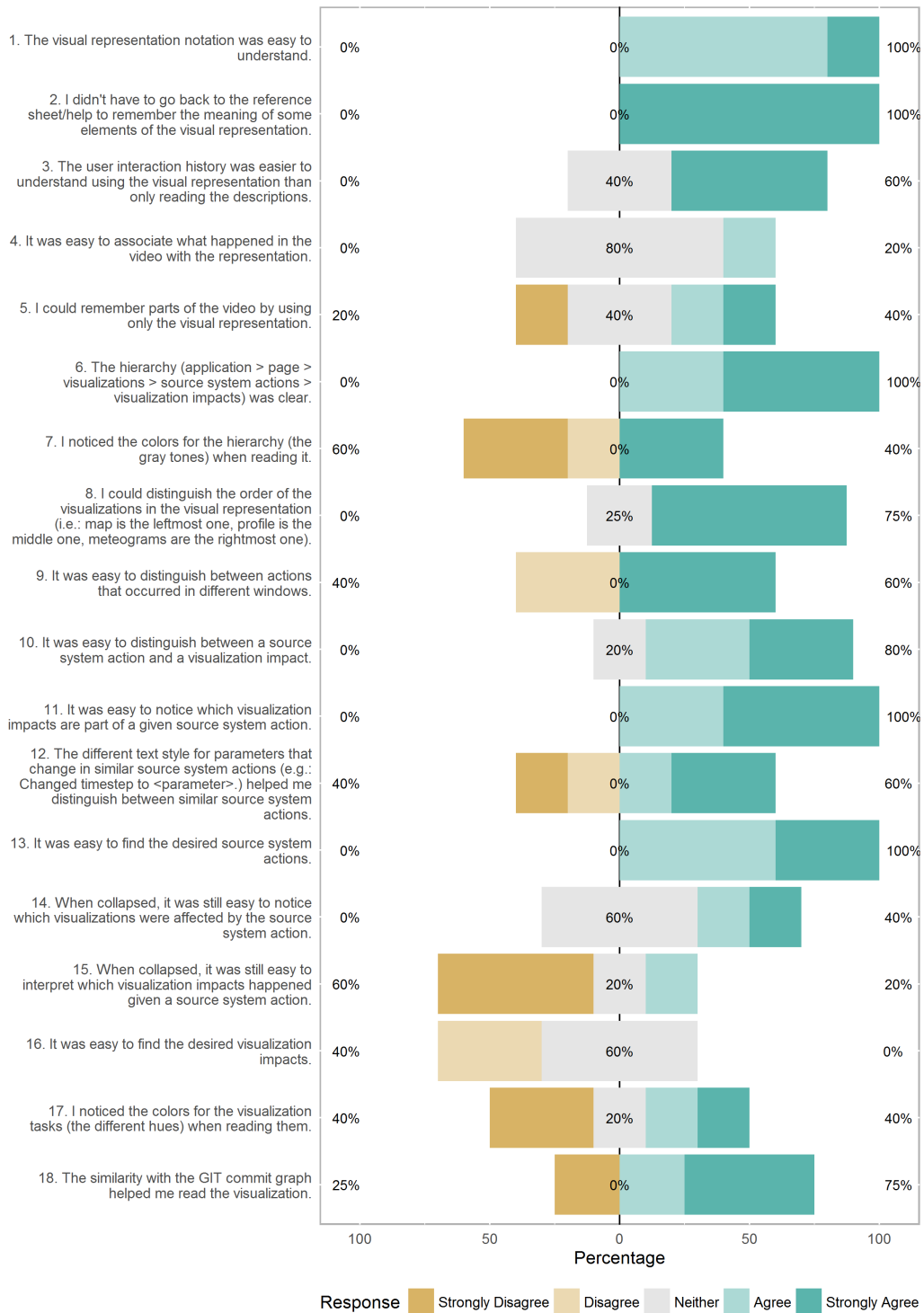


Figure E.9: Distribution of answers to the final questionnaire.

effects, as the latter is the effect caused by the former. Due to the collapse/-expand interaction and banded rows, the participants grasped this concept easily.

Finally, as opposed to task 5 regarding the *visualization effects*, task 4 regarding the *source system actions* had a positive overall outcome (statement 13). We hypothesize that this is due to the construction of the study, focusing on the *source system actions* and allowing the participants to express themselves. When participants found a similar *source system action* description in BONNIE, they were confident about their choice. The same did not happen with the *visualization effects*, since they were organized regarding *visualization components* and *visualization tasks* – concepts particular to BONNIE's log model.

F Narrative Builder User Study Details

After the studies with the history visualization, we wanted to execute a new study considering both parts of BONNIE: the history visualization and the narrative builder. We decided to recruit the same participants from the previous study, conducted about three months before. Put together, the studies can be considered parts of a short-term longitudinal study, focused on different iterations of BONNIE. Once again, the pilot was considered in the results (as participant P0), since the study materials and procedures were the same used with the other participants.

The BONNIE iteration used in this study was the final one shown in figure 3.3. The main differences from the previous study are:

- Different *effect nodes* shapes according to the different *source system actions* types.
- Different *source system actions* descriptions based on the feedback from the previous study.
- Added the possibility of having *source system action* without *visualization effects* (e.g., “Stopped animation at [[timestep]]”).
- Added the *BONNIE annotation action* feature.
- The *visualization tasks* colored tags followed the verb/object approach (instead of single verb), as appears in figure 6.5(c).

This time, the study only had two tasks:

Task 1: Find the visualization states Equal to task 5 from the previous study, we showed a video and asked for the same visualization states. The idea was to compare if the minor modifications already brought some benefits.

Task 2: Create a narrative We asked participants to interact with WISE and generate a narrative afterwards. The idea was to compare if the impact of actually interacting with SrcSys affects the usability of BONNIE.

The next section (section F.1) contains all the material used in the study. The study’s methodology is discussed in section F.2 Section F.3 presents all

the collected study data. A discussion of results can be found in the section 7.1 from the main thesis.

F.1 Study Material

This section presents all material used in the study as follows:

- Figures F.1 to F.3 show the help material used to introduce BONNIE. The participant could use the help material, available online and in print formats, at any moment during the study.
- Figures F.4 and F.5 present the script used to guide the study session.
- Figure F.6 shows the questionnaire regarding the history visualization.
- Figure F.7 shows the questionnaire based on TAM statements.

F.2 Methodology

The study script can be seen in figures F.4 and F.5. It began with an introduction to BONNIE using the available help material (figures F.1, F.2, and F.3). An online and printed version were available at all times to participants. The evaluator explained the help material, answering questions, and inviting the participants to interact with BONNIE at the end.

Following the BONNIE introduction, we repeated the introduction to WISE. We highlighted the main WISE concepts (slide 2), the data characteristics that we explore in the tasks (slide 3), and the main *visualization components* of the UI (slide 4). Due to technical reasons, the meteograms data were not available for this study, so we discouraged participants from using this WISE feature.

Task 1 (slides 5 and 6) asked participants to create a narrative with the same *visualization effects* we asked in the last task (task 5) of the previous study. We used the same exact video from the previous study, narrating it once, and letting the participants watch as many times as needed.

Task 2 (slides 7 to 10) asked participants to interact with WISE, analyzing a rain event from one forecast (observing when it happened, its intensity, and where it happened) and comparing it with the forecast generated the day before. After the open-ended exploration of WISE, the participants should use BONNIE to create a narrative to share their interpretation. We did not evaluate the created narrative, only the usage of BONNIE. Moreover, we asked participants to focus only on the narrative content, given that the

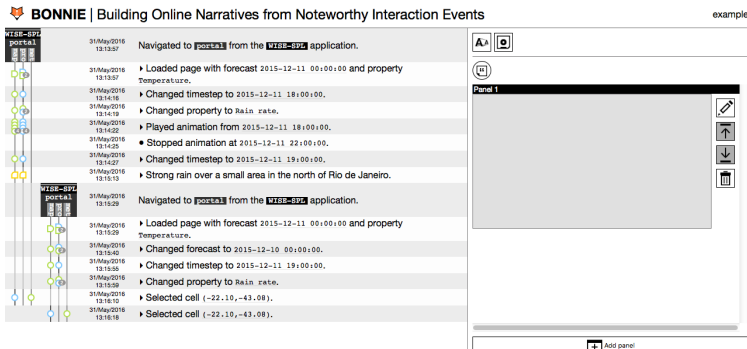
BONNIE | Building Online Narratives from Noteworthy Interaction Events

What is BONNIE?

BONNIE stands for "Building Online Narratives from Noteworthy Interaction Events". The idea is to create a narrative by choosing the relevant steps from previous interaction with another system (called "Source System").

How does it work?

BONNIE can be split into two main parts: the history viewer (on the left side) and the narrative builder (on the right side).



Main UI.


The history viewer shows the steps took while interacting with the Source System. From the history viewer, you can choose the relevant steps to build a narrative. The next sections explain in detail each part and how to build the narrative.

How do I "read" the history visualization?

The history viewer shows the logged interaction events using a graphical representation and a textual description. The events are organized from the oldest one to the most recent one. Reading the textual descriptions from top to bottom would, therefore, follow the events' chronological order.

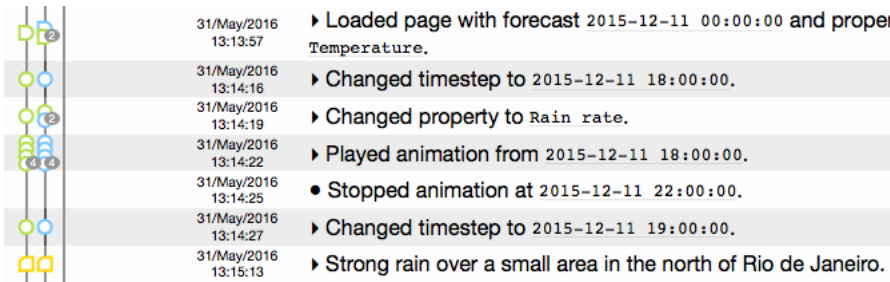
The visualization has three different kinds of rows: navigation, source system action, and visualization impact. The rows' background color are just for legibility, alternating between independent rows.

A **navigation row** is characterized by the break in the graphical representation flow. It shows the **source system**, the **page**, and the page's **visualizations** (the grayscale should help remembering the hierarchy: **source system** contains **pages** that contains **visualizations**). From each visualization name emerges a line that represents how long that given page was active, i.e., there are still actions happening in that page. For example, the image below show 3 different visualizations in that given view: map (left), profile (middle), and meteograms (right).



Navigation row.

A **source system action row** provides a description of the action performed in the source system. It uses a vocabulary specific to each source system and highlights the parameters of each action.



Source system action rows.

A source system action may cause impacts on visualizations (for example, encode new data, change the current highlight). These impacts can be seen as nodes on the representation. A single action may trigger many impacts or none and the nodes' position indicate which visualization was affected. For example, in the figure below, the top action row has caused 8 impacts in total – 4 in

Figure F.1: BONNIE help, page 1.

the first visualization (map) and other 4 in the middle one (profile) -- while the bottom one has no impact on the visualizations -- it kept the same visualization states.

Impacts shown while the source system action row is collapsed. Notice that the bottom row has not caused any impact in the visualizations.

The node shapes indicate different action types (the colors will be explained later):

- Navigation action Used when navigating to a new page to represent the default page loading.
- User action Action performed explicitly by the user in the source system (for example, by clicking on a button).
- System action Action performed automatically by the source system (for example, by automatically downloading new data).
- Comment action While interacting with the source system, user logged a comment to appear in BONNIE.

Finally, **visualization impacts rows** can be seen when expanding a source system action row (all actions start collapsed). When clicking on a source system action row with impacts, they go from collapsed (indicated by the ▶ symbol) to expanded (indicated by the ▼ symbol). The visualization impacts rows become visible, with the same background as its parent source system action row, and the impact nodes "slide" to go to its respective row.

Collapsed source system action row.

Expanded source system action row with its visualization impact rows.

Visualization impacts rows are characterized by the colored tags in the description, sharing the same color with the corresponding impact node. Contrary to the user action row, visualization impacts are domain-independent and categorized according to the following visualization tasks (and colors):

Encoded data	How data is initially encoded as a visual representation (e.g.: load, update)
Selected element	Demarcation of one or more elements in a visualization, differentiating selected from unselected elements (e.g.: select, brush, highlight)
Navigated viewpoint	Alters a user's viewpoint (e.g.: zooming, panning, rotating)
Arranged elements	Organize visual elements (e.g.: reordering axes, rows/columns)
Changed elements	Alterations in visual encoding (e.g.: size and transparency of points, changing the chart type)
Filtered data	Adjust the exclusion and inclusion criteria for elements in a visualization
Aggregated data	Changes the granularity of visualization elements (e.g.: group/split)
Annotated element	Addition of graphical or textual annotations associated with one or more visualization elements (e.g.: comment, note, annotate)
Imported elements	Addition of new elements to the visualization
Derived data	Compute new data elements given existing data elements (e.g.: calculate average, standard deviation)
Recorded visualization	Save or capture visualization elements as persistent artifacts (e.g.: save, print, capture)

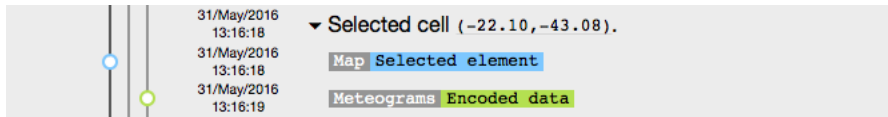
The visualization lines can also hint in which visualization the action occurred (darker segments) and the unrelated pages (lighter segments). For example, from the figure below, we can notice that the first 3 lines are lighter, indicating that this page is unrelated to the action. The 4th and 6th ones are in their neutral color, whilst the 5th one is in a darker tone. This represents that this visualization was related to the action (in this case, the action was changing the timestep by clicking on the profile visualization).

Line colors detail.

When hovering over a source system action row, a comment button appears to edit comments associated with the action row. If an action has comments, the button appears even without hovering showing the number of associated comments.

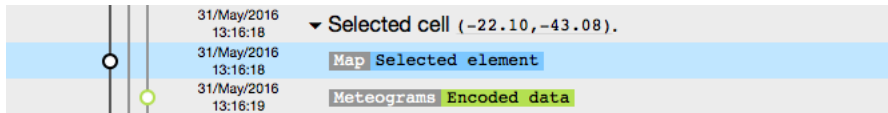
Action row hovering.

Figure F.2: BONNIE help, page 2.

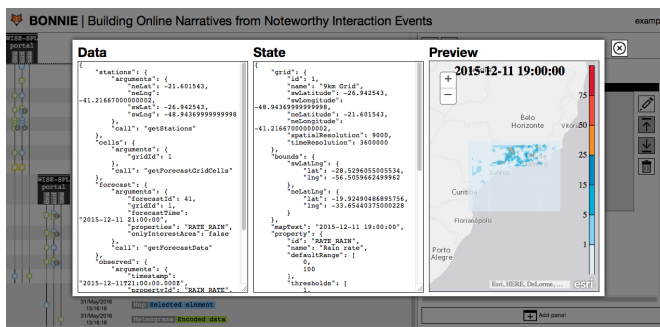


Action row with comment.

When hovering over a visualization impact row, an information button appears to show the metadata associated with that impact. In the popup dialog, it is possible to see developer information regarding the encoded data, the description of the new visualization state (after the impact), and a preview of the visualization.



Visualization impact row hovering.



Visualization impact developer metadata.

How do I build a narrative?

Easy: by drag-and-drop!

BONNIE supports creating a narrative composed by sequential panels (think about comics or slideshow). Each panel may contain any number of elements and may be reordered. To add an empty panel you can click on the 'Add panel' button. To add a panel already with an element, you can drop the element in the 'Add panel' button or in any empty space in the narrative builder area.

An element may be a visualization or a text. To create a text element, you may drag the text element icon or an action row. To create a visualization element, you may drag a visualization impact row or node.

Story builder icons.	
Icon	Description
	Toggle scaling textual elements in the panels.
	Save current narrative as a new one.
	Text element icon. Drag and drop on a panel to add a text element.
	Panel advanced layout / Edit text element.
	Move panel upwards.
	Move panel downwards.
	Remove panel / Remove element.
	Add panel.

After creating a narrative, you can save it and share the link to it. Viewing a narrative does not allow to change it, but you can still interact with the visual elements and the data can be updated depending on the source system's data service.

Figure F.3: BONNIE help, page 3.

Study description

- We are going to continue the last study, interacting with the history visualization to create a narrative from the interaction events
- Take your time, there is no right/wrong answers
- If possible, think aloud, just saying whatever comes to your mind, without attempting to create a coherent discourse.

- Agenda:
 - BONNIE introduction
 - WISE recap
 - Task (BONNIE)
 - Questionnaire
 - Task (WISE+BONNIE)
 - Questionnaire
 - Interview

WISE Recap

- Weather InSights Environment
- Shows weather related information from forecast and observed data
- Each forecast:
 - Is generated **daily at midnight**
 - Comprises **48 hours** of predicted data
 - i.e. a forecast generated on January 1st at midnight predicts data up until January 3rd at midnight
 - Has data for every **1 hour**
 - i.e. the forecasted data timestep is of 1 hour

WISE Recap

- Therefore, for a given timestamp there are **2 predicted data**

WISE Recap

Task 1

Last task from previous study

- You are going to watch the **same** video of a user interacting with WISE (we do not expect you to remember it) as many times as you wish.
- Once you are good to go, you will **not** be able to go back to the video: you will just interact with BONNIE.
- We will ask you to create a narrative based on the interaction shown in the video.

Task 1

Last task from previous study

- Create the following narrative:

Panel 1 Metograms for the 2 nd select cell from the 2015-12-11 forecast for the 2015-12-11 10:00:00 timestep Metograms for the 2 nd select cell from the 2015-12-10 forecast for the 2015-12-11 10:00:00 timestep	Panel 2 Metograms for the 1 st select cell from the 2015-12-11 forecast for the 2015-12-11 10:00:00 timestep Metograms for the 1 st select cell from the 2015-12-10 forecast for the 2015-12-11 10:00:00 timestep
Panel 3 Profiles from the 2015-12-11 forecast for the 2015-12-11 10:00:00 timestep Profiles from the 2015-12-10 forecast for the 2015-12-11 10:00:00 timestep	Panel 4 Maps from the 2015-12-11 forecast for the 2015-12-11 12:00:00 timestep Maps from the 2015-12-10 forecast for the 2015-12-11 12:00:00 timestep

Hint: each panel contains the same visualization at the same timestamp, but from different forecasts.

Figure F.4: Study script, page 1.

Task 2
Interacting with WISE & BONNIE

- Pretend you are a WISE user wishing to evaluate the forecast generated on **2015-11-08**.
- Focusing on the **first 24 hours** of the forecast (*i.e.* from 2015-11-07 22:00:00 to 2015-11-08 22:00:00), try to answer these questions:
 - When was the most relevant **rain** event in the continent?
 - What was the intensity (color) of the rain?
 - Which regions did it affect?
 - Did it affect the **temperature** of the region?

7

Task 2
Interacting with WISE & BONNIE

- Considering the same rain event, analyze the forecast generated on **2015-11-07**.
- Focusing on the **last 24 hours** of the forecast (*i.e.* from 2015-11-07 22:00:00 to 2015-11-08 22:00:00, the same time window), consider these questions:
 - Did the rain event appear on this forecast?
 - Was it at the same time or was it earlier/later?
 - Was it at the same location? Did it affect a greater/smaller area?
 - Was it with the same intensity (color)?

8

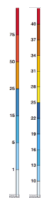
Task 2
Interacting with WISE & BONNIE

- Interacting with BONNIE, create a narrative to share your interpretation of the answers.
- In this task (both while using WISE and BONNIE), we are not evaluating your analysis, but the systems usage. So you do **not** have to be precise.
 - *E.g.*: For the region, you do not have to say any city, just right/left/top/bottom/... For the intensity, you can use the color code.
- Do **not** mind the layout of the narrative panels. Focus only on the content.

9

Task 2
Interacting with WISE & BONNIE

- Hint: Pay attention to the current selected forecast.
- Hint: The rain rate property has fewer levels than the temperature



- Hint: The profile only shows the rain rate distribution.
- Hint: Remember that you can make annotations while interacting with WISE

10

Figure F.5: Study script, page 2.

Task 1 Questionnaire

	1: strongly disagree	5: strongly agree
1. The visual representation notation was easy to understand.		
2. I didn't have to go back to the reference sheet/help to remember the meaning of some elements of the visual representation.		
3. The user interaction history was easier to understand using the visual representation than only reading the descriptions.		
4. It was easy to associate what happened in the video with the representation.		
5. I could remember parts of the video by using only the visual representation.		
In the visual representation:		
6. The hierarchy (application > page > visualizations > source system actions > visualization impacts) was clear.		
7. I noticed the colors for the hierarchy (the gray tones) when reading it.		
8. I could distinguish the order of the visualizations in the visual representation (i.e.: map is the leftmost one, profile is the middle one, meteorgrams are the rightmost one).		
9. It was easy to distinguish between actions that occurred in different windows.		
10. It was easy to distinguish between a source system action and a visualization impact.		
11. It was easy to notice which visualization impacts are part of a given source system action.		
12. The different text style for parameters that change in similar source system actions (e.g.: "Changed timestep to <parameter>") helped me distinguish between similar source system actions.		
13. It was easy to find the desired source system actions.		
14. When collapsed, it was still easy to notice which visualizations were affected by the source system action.		
15. When collapsed, it was still easy to interpret which visualization impacts happened given a source system action.		
16. It was easy to find the desired visualization impacts.		
17. I noticed the colors for the visualization tasks (the different hues) when reading them.		
18. The similarity with the GIT commit graph helped me read the visualization.		

Figure F.6: History visualization questionnaire. Similar to the one from the previous study.

P.S.: Whenever "tasks" or "job" is mentioned, consider using BONNIE integrated with some application useful for you (i.e. not WISE necessarily).

	1: extremely unlikely	2: quite unlikely	3: slightly unlikely	4: neither	5: slightly likely	6: quite likely	7: extremely likely
1. Using BONNIE would enable me to accomplish tasks more quickly.							
2. Learning to operate BONNIE would be easy for me.							
3. Assuming I have access to BONNIE, I intend to use it.							
4. Using BONNIE would improve my task performance.							
5. I would find it easy to get BONNIE to do what I want it to do.							
6. The quality of the output I get from BONNIE is high.							
7. Using BONNIE would improve my task productivity.							
8. My interaction with BONNIE would be clear and understandable.							
9. I would have no difficulty telling others about the results of using BONNIE.							
10. In my job, usage of BONNIE would be important.							
11. Using BONNIE would enhance my task effectiveness.							
12. I would find BONNIE to be flexible to interact with.							
13. I believe I would be able to communicate to others the consequences of using BONNIE.							
14. Given that I have access to BONNIE, I predict that I would use it.							
15. Using BONNIE would make it easier to do my tasks.							
16. It would be easy for me to become skillful at using BONNIE.							
17. The results of using BONNIE would be apparent to me.							
18. I would have no problem with the quality of BONNIE's output.							
19. I would find BONNIE useful in performing my tasks.							
20. I would find BONNIE easy to use.							
21. I would find it easy to explain why using BONNIE may or may not be beneficial.							
22. In my job, usage of BONNIE would be relevant.							

Figure F.7: TAM questionnaire, combining statements from TAM and TAM2.

Table F.1: Association between TAM/TAM2 constructs and the questions.

Source	Construct	Questions
TAM	Perceived usefulness	1, 4, 7, 11, 15, 19
TAM	Perceived ease of use	2, 5, 8, 12, 16, 20
TAM2	Intention to use	3, 14
TAM2	Output quality	6, 18
TAM2	Result demonstrability	9, 13, 17, 21
TAM2	Job relevance	10, 22

layout of panel elements can only be changed currently with the *advanced layout* feature.

After each task, the participants answered two questionnaires. The first was very similar to the one from the previous study (figure F.6), with minor alterations regarding the terminology (*e.g.*, replacing “visualization action” with “visualization impact”) and the task at hand (*e.g.*, replacing “video” with “my interaction”). The second one (figure F.7) followed the original technology acceptance model (TAM) (Davis, 1989) and some constructs of the TAM2 (Venkatesh & Davis, 2000), namely “intention to use”, “job relevance”, “output quality”, and “result demonstrability”.

The 22 questions were based on the TAM and TAM2 standard ones, adapting them to refer to BONNIE and rephrasing them so every statement had a “positive” meaning if the participant agreed with the statement. We also made minor alterations so the TAM2 statements were better phrased to use TAM’s Likert scale. Table F.1 association between constructs and questions.

The idea of answering the questionnaires after each task was to evaluate if the actual interaction with the SrcSys would somehow impact the interaction with BONNIE. To avoid the learning effect of performing task 2 after task 1, we randomized the tasks order (P3 and P4 performed task 2 before task 1) so the learning effect of a user is offset by another one. Consequently, the entire data set is not significantly biased by the learning effect (Lazar et al., 2010, p. 52).

F.3 Study Results

This section presents all collected data in the study, as follows:

- Table F.2 shows the results of task 1, with checkmarks indicating correct answers or the kind of mistake the participant made.
- Table F.3 shows the history visualization questionnaire answers after task 1.
- Table F.4 shows the TAM questionnaire answers after task 1.

Table F.2: Task 1 results.

Item	Forecast	P0	P1	P2	P3	P4
Met 2nd cell pair	2015-12-11	✓	✓	✓	✓	✓
	2015-12-10	✓	✓	✓	✓	wrong cell
Met 1st cell pair	2015-12-11	✓	✓	✓	✓	✓
	2015-12-10	wrong time	wrong time	wrong time	wrong time	✓
Profile	2015-12-11	✓	✓	✓	✓	✓
	2015-12-10	✓	✓	✓	✓	✓
Map	2015-12-11	✓	✓	✓	✓	wrong view
	2015-12-10	✓	✓	✓	✓	✓

- Table F.5 shows the history visualization questionnaire answers after task 2.
- Table F.6 shows the TAM questionnaire answers after task 2.

Table F.3: Task 1 history visualization questionnaire answers by participant.

Statement	P0	P1	P2	P3	P4
1. The visual representation notation was easy to understand.	4	4	5	5	4
2. I didn't have to go back to the reference sheet/help to remember the meaning of some elements of the visual representation.	3	5	1	5	5
3. The user interaction history was easier to understand using the visual representation than only reading the descriptions.	3	5	3	3	3
4. It was easy to associate what happened in the video with the representation.	2	5	3	5	4
5. I could remember parts of the video by using only the visual representation.	3	4	3	5	5
6. The hierarchy (application > page > visualizations > source system actions > visualization impacts) was clear.	5	5	5	5	5
7. I noticed the colors for the hierarchy (the gray tones) when reading it.	2	1	1	5	3
8. I could distinguish the order of the visualizations in the visual representation (i.e.: map is the leftmost one, profile is the middle one, meteograms are the rightmost one).	3	4	5	5	4
9. It was easy to distinguish between actions that occurred in different windows.	5	4	5	5	5
10. It was easy to distinguish between a source system action and a visualization impact.	3	5	5	4	5
11. It was easy to notice which visualization impacts are part of a given source system action.	4	5	5	5	4
12. The different text style for parameters that change in similar source system actions (e.g.: "Changed timestep to <parameter>.") helped me distinguish between similar source system actions.	5	3	5	5	4
13. It was easy to find the desired source system actions.	3	3	3	5	4
14. When collapsed, it was still easy to notice which visualizations were affected by the source system action.	2	4	5	5	4
15. When collapsed, it was still easy to interpret which visualization impacts happened given a source system action.	2	4	5	5	4
16. It was easy to find the desired visualization impacts.	3	3	4	4	4
17. I noticed the colors for the visualization tasks (the different hues) when reading them.	5	2	5	5	5
18. The similarity with the GIT commit graph helped me read the visualization.	5	4	1	4	5

Table F.4: Task 1 TAM questionnaire answers by participant.

Statement	P0	P1	P2	P3	P4
1. Using BONNIE would enable me to accomplish tasks more quickly.	6	5	7	6	7
2. Learning to operate BONNIE would be easy for me.	7	6	7	7	6
3. Assuming I have access to BONNIE, I intend to use it.	5	5	7	7	7
4. Using BONNIE would improve my task performance.	4	4	7	6	6
5. I would find it easy to get BONNIE to do what I want it to do.	5	6	7	7	6
6. The quality of the output I get from BONNIE is high.	4	7	7	7	6
7. Using BONNIE would improve my task productivity.	4	4	6	6	7
8. My interaction with BONNIE would be clear and understandable.	5	5	7	7	6
9. I would have no difficulty telling others about the results of using BONNIE.	6	5	7	7	7
10. In my job, usage of BONNIE would be important.	3	3	2	5	7
11. Using BONNIE would enhance my task effectiveness.	4	4	7	6	7
12. I would find BONNIE to be flexible to interact with.	5	6	7	7	6
13. I believe I would be able to communicate to others the consequences of using BONNIE.	4	4	7	7	7
14. Given that I have access to BONNIE, I predict that I would use it.	3	5	7	7	7
15. Using BONNIE would make it easier to do my tasks.	4	3	7	6	7
16. It would be easy for me to become skillful at using BONNIE.	5	7	5	7	6
17. The results of using BONNIE would be apparent to me.	4	6	6	7	7
18. I would have no problem with the quality of BONNIE's output.	3	7	6	7	6
19. I would find BONNIE useful in performing my tasks.	4	3	7	5	7
20. I would find BONNIE easy to use.	6	7	7	7	6
21. I would find it easy to explain why using BONNIE may or may not be beneficial.	4	5	7	7	7
22. In my job, usage of BONNIE would be relevant.	4	4	2	5	7

Table F.5: Task 2 history visualization questionnaire answers by participant.

Statement	P0	P1	P2	P3	P4
1. The visual representation notation was easy to understand.	5	4	5	5	4
2. I didn't have to go back to the reference sheet/help to remember the meaning of some elements of the visual representation.	3	5	5	5	5
3. The user interaction history was easier to understand using the visual representation than only reading the descriptions.	2	5	5	4	3
4. It was easy to associate what happened in the video with the representation.	4	4	5	5	5
5. I could remember parts of the video by using only the visual representation.	4	4	5	5	3
6. The hierarchy (application > page > visualizations > source system actions > visualization impacts) was clear.	5	5	5	5	4
7. I noticed the colors for the hierarchy (the gray tones) when reading it.	1	2	5	2	3
8. I could distinguish the order of the visualizations in the visual representation (i.e.: map is the leftmost one, profile is the middle one, meteograms are the rightmost one).	4	3	5	5	5
9. It was easy to distinguish between actions that occurred in different windows.	5	3	n/a	n/a	3
10. It was easy to distinguish between a source system action and a visualization impact.	3	5	5	4	5
11. It was easy to notice which visualization impacts are part of a given source system action.	3	5	5	5	5
12. The different text style for parameters that change in similar source system actions (e.g.: "Changed timestep to <parameter>.") helped me distinguish between similar source system actions.	5	4	5	5	5
13. It was easy to find the desired source system actions.	4	3	5	5	5
14. When collapsed, it was still easy to notice which visualizations were affected by the source system action.	2	5	5	5	4
15. When collapsed, it was still easy to interpret which visualization impacts happened given a source system action.	2	5	5	5	4
16. It was easy to find the desired visualization impacts.	4	2	5	5	5
17. I noticed the colors for the visualization tasks (the different hues) when reading them.	5	1	5	5	5
18. The similarity with the GIT commit graph helped me read the visualization.	5	4	1	4	5

Table F.6: Task 2 TAM questionnaire answers by participant.

Statement	P0	P1	P2	P3	P4
1. Using BONNIE would enable me to accomplish tasks more quickly.	6	6	7	7	7
2. Learning to operate BONNIE would be easy for me.	6	7	7	6	5
3. Assuming I have access to BONNIE, I intend to use it.	4	5	7	7	6
4. Using BONNIE would improve my task performance.	4	4	7	7	7
5. I would find it easy to get BONNIE to do what I want it to do.	7	6	7	7	6
6. The quality of the output I get from BONNIE is high.	5	7	7	7	6
7. Using BONNIE would improve my task productivity.	4	4	6	7	6
8. My interaction with BONNIE would be clear and understandable.	6	7	7	6	6
9. I would have no difficulty telling others about the results of using BONNIE.	6	6	7	7	7
10. In my job, usage of BONNIE would be important.	4	3	2	5	7
11. Using BONNIE would enhance my task effectiveness.	4	3	7	6	7
12. I would find BONNIE to be flexible to interact with.	6	5	7	7	5
13. I believe I would be able to communicate to others the consequences of using BONNIE.	5	4	7	7	7
14. Given that I have access to BONNIE, I predict that I would use it.	4	4	7	7	7
15. Using BONNIE would make it easier to do my tasks.	4	3	7	6	7
16. It would be easy for me to become skillful at using BONNIE.	6	6	5	7	6
17. The results of using BONNIE would be apparent to me.	7	5	6	7	7
18. I would have no problem with the quality of BONNIE's output.	6	7	6	5	6
19. I would find BONNIE useful in performing my tasks.	5	4	7	6	7
20. I would find BONNIE easy to use.	6	6	7	7	6
21. I would find it easy to explain why using BONNIE may or may not be beneficial.	5	5	7	7	7
22. In my job, usage of BONNIE would be relevant.	4	5	2	5	7

G Backlog

This appendix lists some features not yet implemented in BONNIE. Many of these features would be necessary for its wide usage. The list is organized in two hierarchy levels, similar to stories and tasks in an agile development approach.

- Changes to the history visualization
 - Display session navigation in the graph
 - Basic search functionality
 - Group trails by session
 - Playback history
 - Highlight similar VisStates
- Saving stories
 - Save JSON in a DB instead of local files
 - Warn if a story is being overwritten
 - “Save” action (reuse the project name)
 - Load a saved story
- Narrative builder features
 - Pre-defined layouts for panels
 - Reorder panels with drag-and-drop
 - Move panel elements with drag-and-drop
 - Resize panel elements with drag-and-drop
 - Create panel element from any visualization component, even without an associated visualization effect
 - Undo/Redo
 - Split panel into multiple ones
 - Merge panels into a single one
- Log size selector
 - Show a histogram of log activity
 - Select the timespan of log data using the histogram
- Annotations

- Allow pasting an image as a VApp annotation
- Advanced filtering
 - Hide a row
 - Hide/Collapse all actions related to a view
 - Hide/Collapse all actions from a column
 - Filter by visualization effect task
 - Filter by user action type
 - Filter by visualization effect associated visualization component
- User history log
 - Use WebSocket for the logger API
 - Reuse VisState
 - Reuse Data
 - Save a thumbnail of the visualization component